TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

PABLO PASCUAL CAMPO

ANTENNA STEERING SYSTEM FOR DIRECTIONAL MICROWAVE LINK WITH UAV COMMUNICATIONS

Master of Science thesis

Examiner: Dr. Alexander Pyattaev and
Prof. Evgeny Kucheryavy
Examiner and topic approved by the
Faculty Council of the Faculty of
Tampere University of Technology
on January 31, 2018

ABSTRACT

PABLO PASCUAL CAMPO: Antenna steering system for directional microwave link with UAV communications
Tampere University of Technology
Master of Science thesis, 67 pages, 26 Appendix pages
March 22, 2018
Master's Degree Programme in Electrical Engineering
Examiner: Dr. Alexander Pyattaev and Prof. Evgeny Kucheryavy
Keywords: UAV, Steering, Tracking, Directional, Antenna, Drone, Backbone Link, Backhaul link, Servo motors

In the past few years, new uses for UAV (Unmanned Aerial Vehicles) have been discovered and deeply developed due to the potential advantages this technology can offer in the modern world. One attractive use for this aerial technology is the carriage of LTE stations to provide temporary wireless coverage in certain scenarios.

Necessary information for the proper operation of the system is exchanged through a key backbone link established between UAV and a ground station (GS). However, with current designs, this link can only be established for small distances. For many applications concerning the use of UAVs, a backbone link working at medium-high distances could be really useful. The goal of this thesis is designing a steering algorithm to make possible a future enlargement of this key link.

To make this possible, two high gain, high directivity antennas are used. The first one will be installed in the UAV, and the second one in the ground station. Behind these devices, a steering system will be integrated to point them properly to each other. This way, the link will be operative all the time.

Along this thesis, the design of the software algorithm providing the angles to point the antennas have been developed. It is also explained the configuration of the servo motors to provide the desired angles. As a result, it is obtained an accurate steering system providing the values of elevation and azimuth for both antennas. It is only left to attach the servo motors to a mechanical structure to obtain the final rotation.

With the steering system developed within this research thesis, many kinds of amateur and professional aerial vehicles can be operated from medium-high distances. The system provides a complete autonomous solution, avoiding the need of any kind of human iteration, which makes the method transparent for the final user.

PREFACE

The motivation for this research thesis originally stemmed from my passion for exploiting new applications and practises regarding UAV technology. Since I was younger, I have always loved remote-controlled devices, it just fascinated me how something could be flown and smoothly controlled from that far away just by means of electromagnetic radiowaves. This is what led me to study the career of IT & Telecommunications, allowing me to keep developing my skills in this research topic and put all my efforts in creating new proposals and solutions that can help scientific research and have an impact in the world.

As a fact, I could not have accomplished my actual level of knowledge without a solid support group. First of all, my parents, Pilar and Jordi, who supported me with love and affection and have provided patient advice and guidance. And secondly, my friends and companions, each of whom has morally supported me throughout the research process.

Thank you all for your adamant support.

Tampere, March 22, 2018,

Pablo Pascual Campo.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF PROGRAMS

LIST OF ABBREVIATIONS AND SYMBOLS

UAV	Unmanned Aerial Vehicle
D2D	Device to device
GPIO	General Port Input Output
GPS	Global Positioning System
GS	Ground Station
GSA	Ground Station Antenna
GUI	Graphical User Interface
HAP	High Altitude Platforms
IC	Integrated Circuit
LAP	Low Altitude Platforms
LOS	Line Of Sight
MIMO	Multiple Input Multiple Output
PCB	Printed Circuit Board
PDF	Probability Density Function
PoE	Power Over Ethernet
PPDR	Public Protection Disaster Relief
PWM	Pulse Width Modulation
RC	Radio Controller
SSH	Secure Shell
WP	Waypoint

ϕ_1	Latitude of the UAV in radians
ϕ_2	Latitude of the ground station antenna in radians
λ_1	Longitude of the UAV in radians
λ_1	Longitude of the ground station antenna in radians
ξ_1	Elevation angle of the ground station antenna
θ_1	Bearing angle of the ground station antenna
ξ_2	Elevation angle of the UAV
θ_2	Bearing angle of the UAV
DC	Duty Cycle
hp	Horse Power
PW	Pulse Width
PWM	Pulse Width Modulation
R	Radius of The Earth

1. INTRODUCTION

1.1 Background and motivation

Nowadays, unmanned flying technology is evolving following giant steps thanks to the possibilities new technology is offering. This recently-unveiled flying vehicles are beginning to be used in many different fields and applications, as enterprises and companies realize the potential advantages of using them to perform its labours. It is because of that many new functionality and possibilities regarding unmanned aerial vehicles (UAV) are starting to be discovered and exploited.

As authors in [1] explain, UAVs are beginning to be used today for many different applications, such as delivery tasks, Public Protection and Disaster Relief (PPDR) missions such as fire or maritime surveillance, vigilance affairs, cartography orders, agriculture handling, filming jobs, etc., and they are used in more and more applications as time goes by [2]. This technology will become in the future a key feature for many enterprises to perform their daily tasks, offering cheap and easy solutions.

One new possible use for this technology is to use a UAV, or a fleet of them [3] [4], placed optimally along the space [5] to act as flying LTE stations [6]. The flying vehicles will therefore provide temporary coverage to users in certain key scenarios [7].

For this purpose, the UAVs typically carry an antenna to establish the proper communication link with the users. Additionally, another antenna is mounted to provide the connection with a ground station [8].

General goals of the project

Before starting with the proper contents of this research thesis, a brief explanation of the overall project as a whole is presented.

For the project, it is proposed to use an UAV as a base station, providing wireless coverage to the final users. The system will carry two sub-links within the backbone link established between the ground station and the aerial vehicle. Let's name the communication link with the users 'Service Area link', and the communication link

with the ground station 'Control link'. These two links are working together to provide the final service.

The mission of the service area link is to receive and provide the data and media requested by the users, while the purpose of the backbone link is to establish a reliable and safe communication with the ground station. The ground station is connected to a base station, communicating at the same time within the core network.

At this point, one may have guessed the need of a routing device integrated in the UAV to address the packets according to their content: Telemetry packets are delivered to an on-board computer integrated in the UAV, and media packets are sent through the service area link to reach the users, as proposed in [9].

The main configuration thought for the system is shown in Fig. 1.1. Note that both service and control link are allocated within the backbone link.

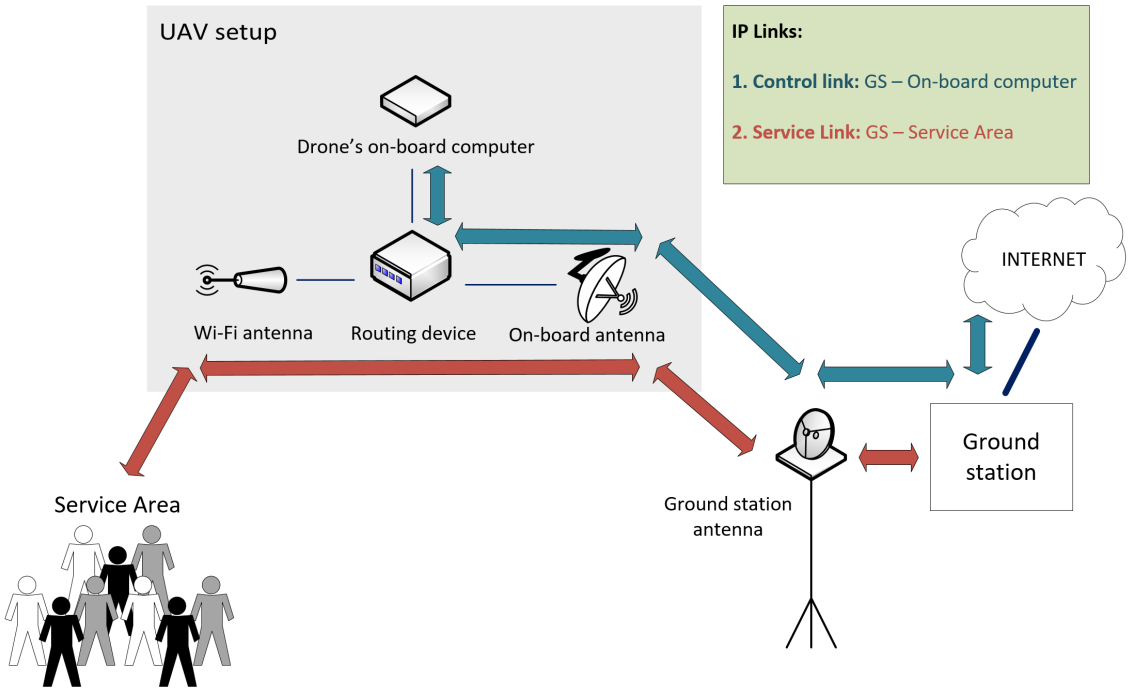


Figure 1.1 *An illustration showing the overall configuration of the system, where both sub-links are allocated within the backbone link.*

Contribution of this research thesis

Once described the general idea of the project as a whole, let's focus on the work, contribution, and motivation of this research thesis.

One attractive feature to implement in the backbone link shown in Fig. 1.1 is to make it work at large distances between ground station and UAV. By having this

configuration, the aerial vehicle can be controlled without the necessity of moving the ground station. This will give more freedom to the UAV, capable of moving around the area.

As an approach to this setup, high directive antennas in both UAV and ground station are installed. These directional antennas concentrate the power radiated in a very narrow region of the space, allowing us to maximize the distance of the backbone link established.

However, this configuration requires both devices to be pointed to each other at all time. As the antennas are very directional, having a mispointing of just a few degrees can strongly degrade the signal received.

At this point is when it comes the necessity of a precise tracking algorithm pointing both antennas to each other. This system needs to be autonomous and fully integrated with both ends.

The algorithm will be installed in both UAV and ground station ends and will communicate and synchronize both devices in real time. The study, design and implementation of the steering algorithm is the final goal of this document, where the fully process to its success it is going to be explained and detailed.

Additionally, it has also been configure a set of servo motors to provide the desired rotation angle. These devices will provide the final elevation and bearing angles according to the steering algorithm.

1.2 Structure of the document

To illustrate the process, the document has been divided in six chapters, explaining the procedure progressively.

The first chapter presents an introduction to the project, to make the reader understand the final purposes and goals of it and how to achieve them. In this first chapter, the author also presents a brief state of art section of this field of study to understand the general background of these new technologies.

At the end of the chapter, two use cases containing two examples of specific scenarios in which our proposal can be used are depicted. The usage of the system is well explained for both of them, comparing our proposal with solutions that are currently used for this applications. The scenarios are extracted from a poster published by the author (see Appendix A.1) in the conference TAKE 5G, held in Helsinki

on 14/12/2017.

The second chapter contains all the pre-configuration and previous setup to configure properly the entire system. The author explains how to configure the communication between the flying vehicle and the ground station.

The third chapter tackles the main goal of the project, the antenna steering system for both UAV and ground station. Once configured all the networks/subnetworks and all the hardware regarding UAV and ground station, it is time to start developing the proper tracking system.

The chapter contains six sections that goes progressively through the design process of the steering system, explaining in detail every step made by the author.

The author first presents the parameters needed to compute the angles for the elevation and bearing to point the antennas, and how to obtain them from the UAV and ground station.

In order to compute these four angles (two for each antenna), a set of parameters between the UAV and the ground station are exchanged. To make this possible, a communication link in real time between both ends is established. The procedure followed by the author to do so is explained in this section.

Finally, after having the final angles of elevation and bearing in both ends, the author shows how to point properly the servo motors for both UAV and ground station antenna (GSA). The configuration process followed is explained within this section.

The fourth chapter contains a section where simulations and tests have been carried by the author out to verify the operation of our steering algorithm and an estimation of its potential error.

The fifth chapter contains a section regarding future work and next steps for the project proposed by the author. We should not forget that the implementation made during this Master's Thesis is still a prototype, and there are a lot of aspects and features to be improved. As a consequence, this section presents several ideas to improve the functionality of the project in a close future. There are presented as well some completely new proposals to be implemented in the project.

The sixth chapter contains the conclusions of the author. This section contains a brief summary of the main ideas presented within the Master's thesis work.

Finally, a chapter containing the bibliography used by the author to make possible the writing of this document is presented.

1.3 Example of two real use cases

The final purpose of the overall project is to provide a secure wireless backbone link to backup the communications within the core network. By improving the hardware integrated in the UAV, the link can be extended to work in medium-high distances, which makes it very useful in many situations.

In this section there are presented two use cases in which our project and idea could be used for.

One use case for the project could be when great events are held and the core network is not capable of providing enough throughput for all the users. For this sake, the UAV or a set of them could be placed to backup the communication link with the core network.

Another use case is when a natural disaster occurs. Our UAV can be deployed in the area to course emergency calls or emergency messages. This system could also help in rescue missions carried out in open sea by establishing a communication link with base stations placed in the shore.

Authors in [10] study coverage aspects of low-altitude platforms for disaster areas, urban and rural. In [11] and [12] authors propose a statistical propagation model for air-to-ground path loss between low altitude platforms (LAP) or high altitude platforms (HAP) and a terrestrial terminal. Solutions with balloons are not usually very fast a flexible because of slow movement and large dimensions, for this reason aerial base stations using drones with propellers specifically quadcopters, hexacopters or octocopters are also used by other groups.

In the following lines, two concrete scenarios are presented in order to show the potential advantages of the system. Both scenarios are extracted from a poster published by our team and I (see Appendix A.1) in the workshop TAKE 5G, held in Helsinki on 14/12/2017.

1.3.1 Scenario 1: Musical event

The first scenario proposed is a big concert held in a remote area. Let's consider that in this musical event there will be a large number of people connecting to Facebook,

Instagram, Twitter... They will most probably be uploading and downloading media content such as photos, videos, audio recordings, etc. These media content have heavy payloads and, therefore, the network will be incapable to course all of this sudden internet traffic.

Now is when the UAVs or a fleet of them go in to backup the backbone link with the core network. By deploying the aerial vehicles, the network will highly increase its throughput, reducing significantly the delays for the final users. This scenario is depicted in Fig. 1.2.

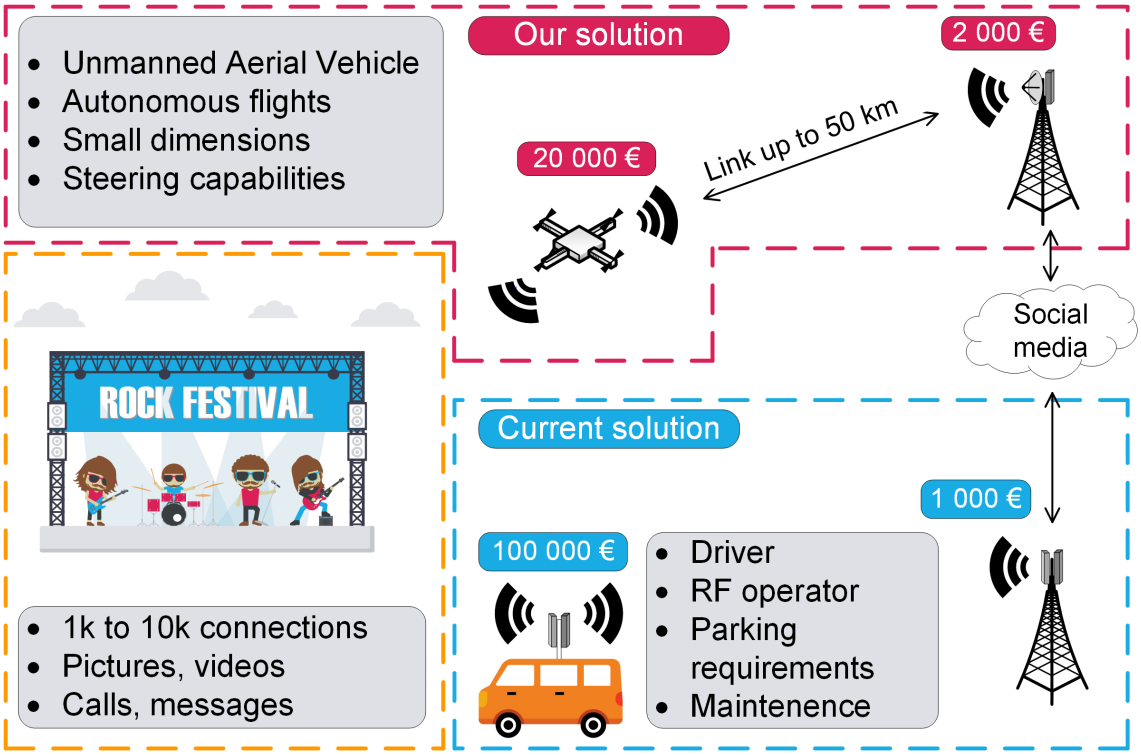


Figure 1.2 *First example of use case: musical event held in a remote location with poor or even no base station coverage.*

Currently solutions to this problem involve the use of a truck with a mounted antenna to provide the link for the communication. As it is showed in Fig. 1.2, the costs of using this solution are extremely high compared with the one proposed in this document.

The authors in [13] study the coverage of these flying base stations in urban scenarios and obtain the optimal altitude of the aerial LTE station to provide maximum coverage in the area. This results or other similar alternatives could be used to control the UAV in this use case.

1.3.2 Scenario 2: PPDR in a flood natural disaster

The second use case considered is related to PPDR. In this specific example, it is presented an area affected by a flood. These kind of scenarios are studied by authors in [14], where wireless coverage for this areas is provided by low altitude aerial platforms, a solution similar to the one studied in our project.

When a natural disaster occurs, it is likely for base stations to be not operative or even lying down in the ground. When this happens, people inside the affected area will not be able to establish communication with other regions.

Device to Device (D2D) connection in PPDR scenarios is also studied in [15], where the connection between mobile platforms with no need of a base station to offer coverage in this kind of scenarios is analyzed.

The aerial vehicles can be deployed in these affected regions to provide the necessary link to communicate with another operative base station (see Fig. 1.3). With this configuration, the UAV can course emergency calls to the core network.

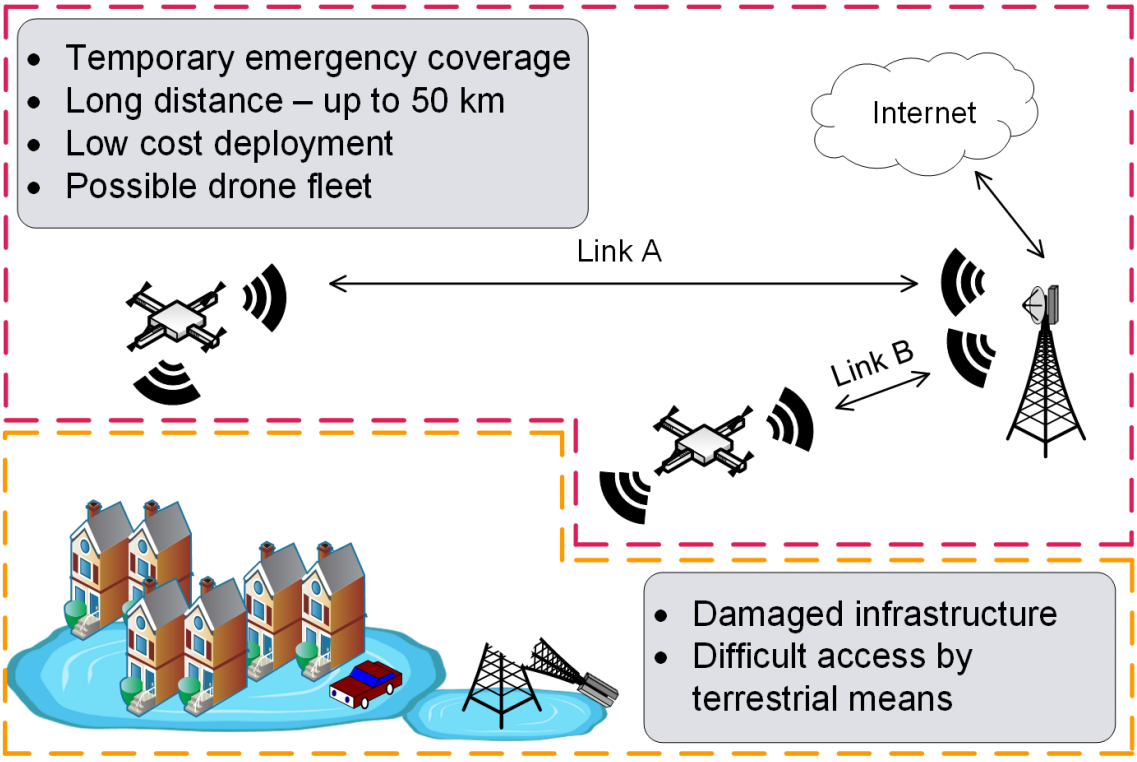


Figure 1.3 *Second example of use case: PPDR in a flood scenario where no base station coverage is available.*

This use case can also be extended to many other situations, such as areas affected by hurricanes, tsunamis, etc.

2. NETWORK PLAN AND BACKBONE LINK CONFIGURATION

Before starting the disign process of the proper steering algorithm, it is necessary to build a network plan for the system. This initial plan will allow to start configuring the devices and to get them ready to communicate to each other, vital to start developing our algorithm.

In this chapter, the configuration of the backbone link to provide the desired communication between both ends is explained, along with the hardware used to make the link work properly.

2.1 Backbone link

As it has been presented in Section 1.1, two main links are required to handle the data exchanged trough the network.

- Control link: To manage telemetry messages for the UAV.
- Service link: To manage user's data.

These two communication links can be allocated in the same backbone link, stablished between UAV and ground station. In order to offer independence between user's data and telemetry messages, the network can be divided into several subnetworks.

During this research thesis, we are focusing in the control link (telemetry link), necessary to interchange parameters between both ends of the communication.

The network address plan chosen for the system is shown in Fig. 2.1.

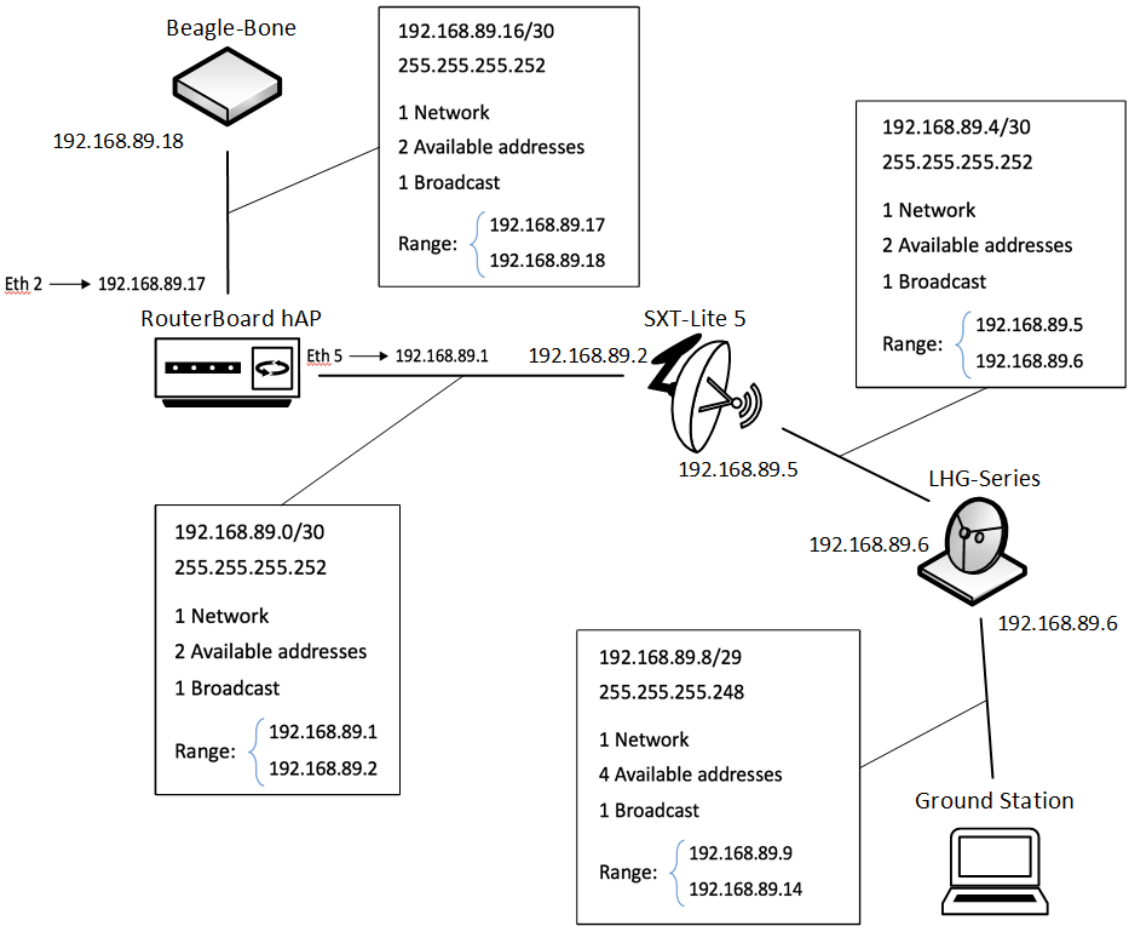


Figure 2.1 *Network address plan configured for the system.*

The different sub-networks used for the system are:

- 192.168.89.0/30 (255.255.255.252): Network for interconnecting the routing devices allocated in the UAV's end.
- 192.168.89.4/30 (255.255.255.252): Network for interconnecting the antenna mounted in the UAV's end with the ground station antenna.
- 192.168.89.8/29 (255.255.255.248): Network for interconnecting the routing devices allocated in the ground station.
- 192.168.89.16/30 (255.255.255.252): Network for transmitting telemetry packets to the UAV's end.

In the future, a firewall can be configured in the routing device to avoid intentional harmful messages that could damage the network.

2.1.1 Telemetry and control link

As explained above, the backbone link will carry the telemetry link to exchange messages between UAV and ground station. The telemetry link will contain the orders and commands that are being used to control the aerial vehicle.

To establish this link, it is required the installation of a ground control station in the ground station and UAV. By doing this, an easy connection between both devices can be established. The messages exchanged through this telemetry link will contain useful parameters for the algorithm development, so it is necessary to fully configure it.

For the ground control station, MAVProxy and MAVLink are the softwares chosen. A brief explanation of them is presented in the followingl lines.

MAVProxy

MAVProxy is a ground control station that can be installed and operated with UAVs. This open-source software supports functionality for different types of vehicles such as copters, planes or rovers, as presented in [16].

MAVProxy is a functional tool in the ArduPilot ecosystem to operate remotely aerial unmanned vehicles. The final purpose of Mavproxy is to make an easy and portable ground control station to command the target UAV. The vehicle should have installed MAVLink protocol in order to be interconnected with the control station.

The software contains a command-line with the possibility of installing a graphical user interface (GUI). It can be run with any device implementing Linux, OS X or Windows.

One key functionality of this software is the possibility to load custom python modules to the software. This is very important to implement our own functions, such as the steering algorithm and servo motor configuration, final goal of the thesis.

MAVLink

MAVLink (Micro Air Vehicle Link) [17] is a protocol which can be used by MAVProxy to communicate UAVs with the ground control station. This protocol handles all the messages necessary to perform the proper communication between the ground control station and the target UAV, as presented in [18].

MAVLink uses a sequence of structured and codified bytes that can be sent to the copter wirelessly (for our use case). The packets sent by this protocol have the structure presented in Fig. 2.2.

Figure 2.2 *Structure of a MAVLink message bit by bit, transmitted within MAVProxy.*

The packet contains the fields of length, sequence number and checksum. Along with this, the message has different ID fields to be identified uniquely:

- System ID: Allows the differentiation of MAVProxys within the same network.
- Sender ID: Allows the differentiation of components within the system.
- Message ID: Allows the identification and decodification of the content.

The structure of the message and specifically the structure of the payload is analyzed in deep detail in Section 3.2, where the extraction of specific content from this field is needed for the algorithm.

2.2 Configuration of the link: Hardware devices

In this section, it is presented a brief description of the devices used in the network. The configuration and use case for each one is shown along with its limitations.

2.2.1 Mikrotik RouterBoard hAP

This device is a Mikrotik branded router[1] mounted in the UAV's end to provide the main routing to telemetry packets, necessary for the steering algorithm.

The main functions to be performed by this device are:

- Connection with the ground station to route the telemetry packets between UAV and ground station.

[1]See "Mikrotik RouterBoard hAP datasheet", 2017: https://i.mt.lv/routerboard/files/hAP-170929154108.pdf

- Connection with BeagleBone to send telemetry packets to be further processed.

As an additional feature, the device has power over ethernet (PoE) ethernet ports, which will be very useful at the time of feeding the devices in the hardware setup [19]. Before mounting the router in the UAV, the external case has been replaced with a lighter one in order to reduce its weight, thus improving the battery life and flight time.

2.2.2 Mikrotik SXT Lite 5

The antenna chosen to be mounted in the UAV's end is the Mikrotik SXT Lite 5[2]. The SXT is a 16 [dBi] antenna working in the band of 5 [GHz] with 20/40 [MHz] channel width to provide outdoor wireless communication. The Wi-Fi protocols supported is IEEE 802.11 a/n.

The mission of this antenna is to communicate with the ground station's end to establish the proper backbone link, where telemetry packets will be exchanged.

The SXT is a directive antenna to provide a large range communication for the backbone link, providing secure levels of SNR at high distances.

The radiation patterns of the on-board antenna are shown in Fig. 2.3.

Note the high directivity of the device, making it necessary to have a steering algorithm to keep both antennas pointed at all time.

2.2.3 Mikrotik LHG-Series

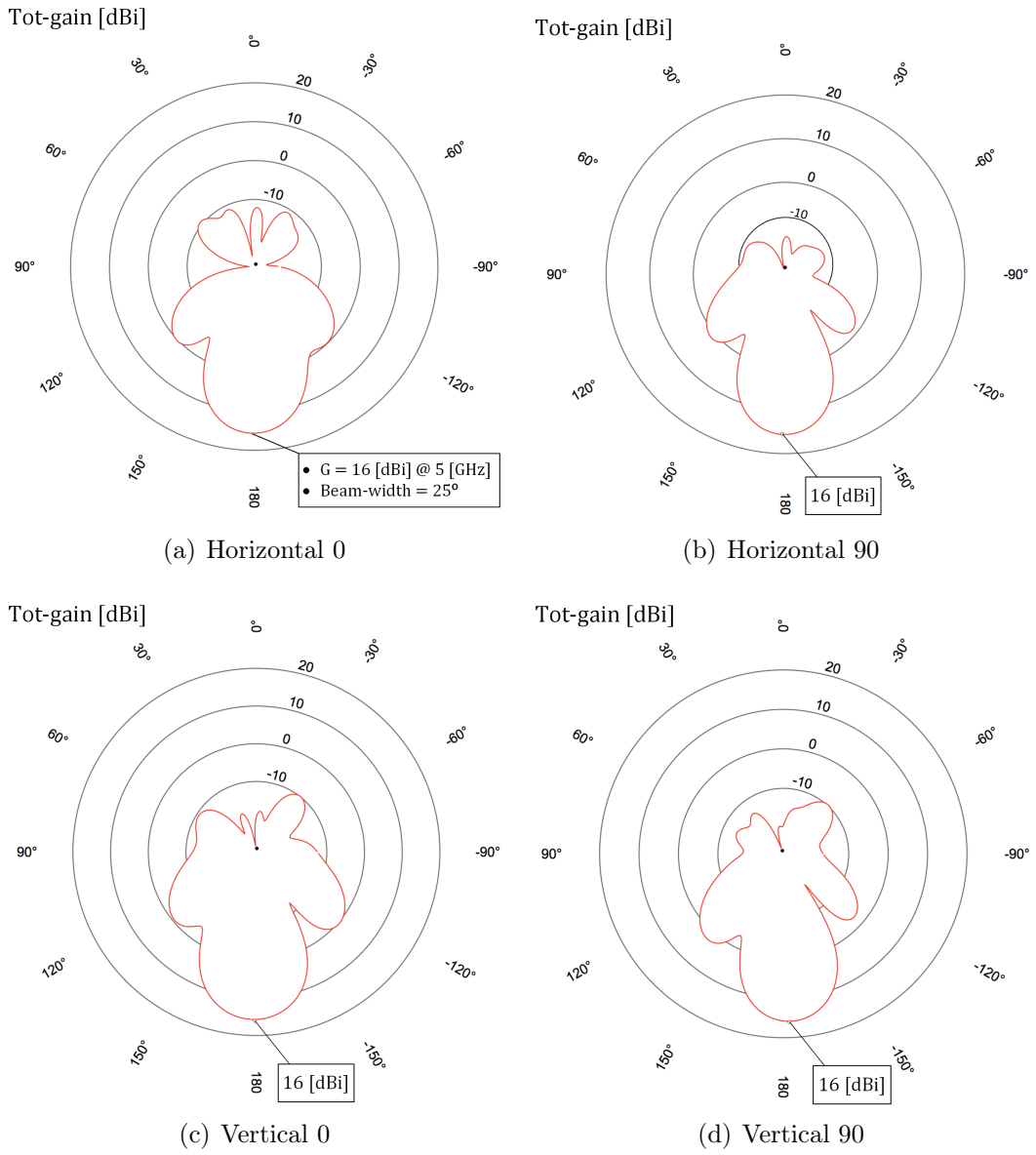
Mikrotik LHG-Series[3] is the device chosen to act as the ground station antenna. It is a directive antenna with a gain of 24.5 [dBi] in the direction of the main lobe. As expected, the device is working in the band of 5 [GHz] with 20/40 [MHz] channel width to communicate with the antenna in the UAV. The protocols supported for the Wi-Fi communication is IEEE 802.11 a/n.

The antenna is physically connected to the ground station and it is responsible of establishing the backbone link to transmit and receive the telemetry packets.

The radiation patterns of the GSA antenna are shown in Fig. 2.4.

[2]See "Mikrotik SXT Lite 5 web datasheet", 2017: https://mikrotik.com/product/RBSXT5nDr2

[3]See "Mikrotik LHG Series datasheet", 2017: https://i.mt.lv/routerboard/files/LHG-170927115805.pdf



(a) Horizontal 0 (b) Horizontal 90

(c) Vertical 0 (d) Vertical 90

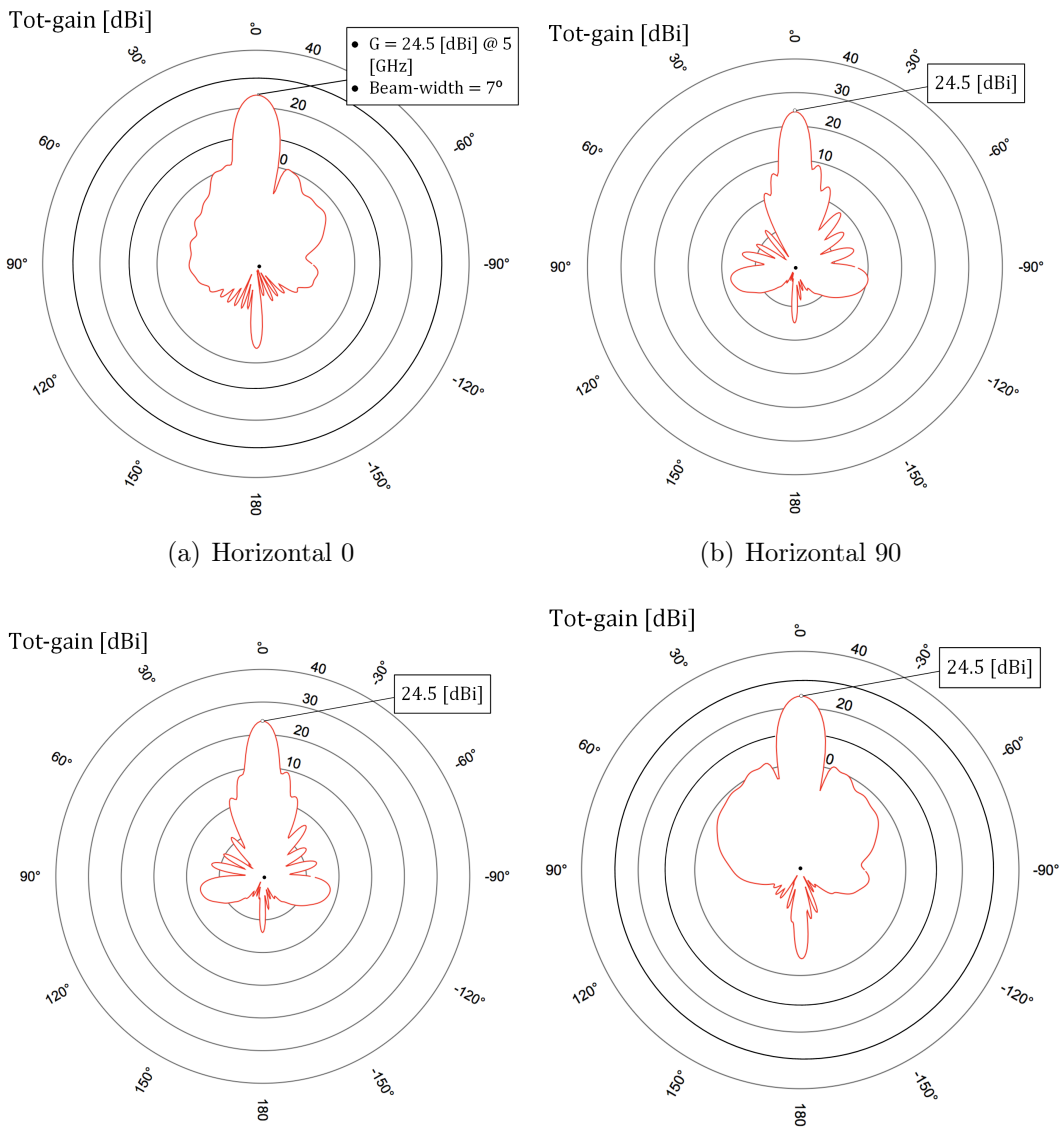
Figure 2.3 *Radiation pattern of the UAV on-board Mikrotik SXT Lite 5 antenna. These plots are extracted from the corresponding datasheets with the permission of Mikrotik.*

As commented before, note the high directivity of the antenna, which requires a steering algorithm if operated with UAVs to track the flying mobile.

2.2.4 BeagleBone

This little and low-power consuming device[4] is installed in the UAV's end to handle the incoming telemetry packets. The BeagleBone is connected directly to the router to receive the control packets coming from the ground station. Its mission is to

[4]See "BeagleBone web datasheet", 2017: http://beagleboard.org/bone



(a) Horizontal 0 (b) Horizontal 90

(c) Vertical 0 (d) Vertical 90

Figure 2.4 *Radiation pattern of the device acting as the ground station antenna, Mikrotik LHG Series, connected physically to the ground station. These plots are extracted from the corresponding datasheets with the permission of Mikrotik.*

traduce and deliver them to the flight controller module. This hardware module is the one handling the movement of the motors of the aerial vehicle to control its trajectory [20]. The BeagleBone is accessed remotely from the ground station through the Backbone-link and operated from there. To do so, it is needed the use of the open softwares Mavproxy and Mavlink (see Section 2.1.1), that allow to emulate a control ground station to command the desired UAV.

3. ANTENNA STEERING ALGORITHM

An antenna steering algorithm is one key feature that could provide a great functionality to this particular system. The final goal of the project is to maximize the distance at which the backbone link is going to operate.

This solution for the steering system is chosen instead of others due to its easy deployment in the aerial vehicle, its potential to satisfy the final goals of the project, and its low cost compared to other solutions.

As far as the author knows, only ground tracking systems have been designed to work with UAVs. In these solutions, only one antenna is configured to keep the track of the aerial vehicle, hence increasing the signal level in it. This antenna is known as ground station antenna, which is typically connected to a ground station to control the UAV.

Our novel implementation also includes a second antenna integrated in the aerial vehicle. This way, all the power delivered by the ground station antenna can be further received with this second device in the UAV, enhancing the signal reception.

As a comparison between both methods, the one implemented during this thesis provides better results regarding power received and SNR in the RX chain of the UAV, allowing the link to work at higher distances, but its implementation is more challenging, as we are considering one additional antenna.

One main difference between both systems is the complexity of the algorithm to be developed to point both antennas. In the scenario where only one ground antenna is needed, the algorithm should just compute the angles to aim this antenna as a function of the position and altitude of the UAV, but with our solution, the new algorithm needs to be run at the same time in the ground station and UAV. The main implication of this is the need of an intelligent device (i.e., BeagleBone) integrated in the UAV and perfectly synchronized with the ground station.

The pointing or steering algorithm of the antenna is well described in the following chapter. Firstly, a section explaining the computation of the angles of elevation and

bearing for both antennas is presented.

After this, it is explained how to extract the needed parameters from MAVLink packages, along with the synchronization carried out by the ground station and the UAV to move the servo motors in the desired angles.

After having all the functions programmed and ready, it is necessary to integrate them within the kernel code of MAVProxy to create a new module with the new functionality.

This process to be followed can be seen as an infinite loop, repeating indefinitely in order to keep the antennas steered. This loop is similar to the one proposed by M. I1amathi and K. Abirami in [21], where a video tracker for a battle tank is studied. The flow of the program can be stated as shown in Fig. 3.1.

3.1 Computation of the elevation and bearing angles

In an antenna tracking system, the final parameters to be obtained are the elevation and bearing of the antennas to track. With these two angles, the devices can be rotated properly pointing to the right angle.

The elevation corresponds to the angle to apply to the ground station antenna in vertical direction with respect to the ground. When talking about the UAV, it refers to the vertical angle to rotate the antenna with respect the horizontal line traced by the UAV when flying. The bearing refers to the angle to apply to the antenna in the horizontal direction with respect to the north.

To tackle the steering system, it is important to understand the parameters that are being used to compute the final results for the elevation and azimuth angles. As authors mention in [22], by having the longitude, latitude, altitude and heading in the Earth-fixed coordinates of the target UAV; the final angles to steer the antennas can be obtained. As we are adding another antenna in the UAV's end, we need as well the longitude, latitude and altitude of the ground station antenna.

As a consequence, the input/output parameters to be manipulated are:

- Input parameters:
  - $\phi_1 \rightarrow$ Latitude of the UAV.
  - $\lambda_1 \rightarrow$ Longitude of the UAV.

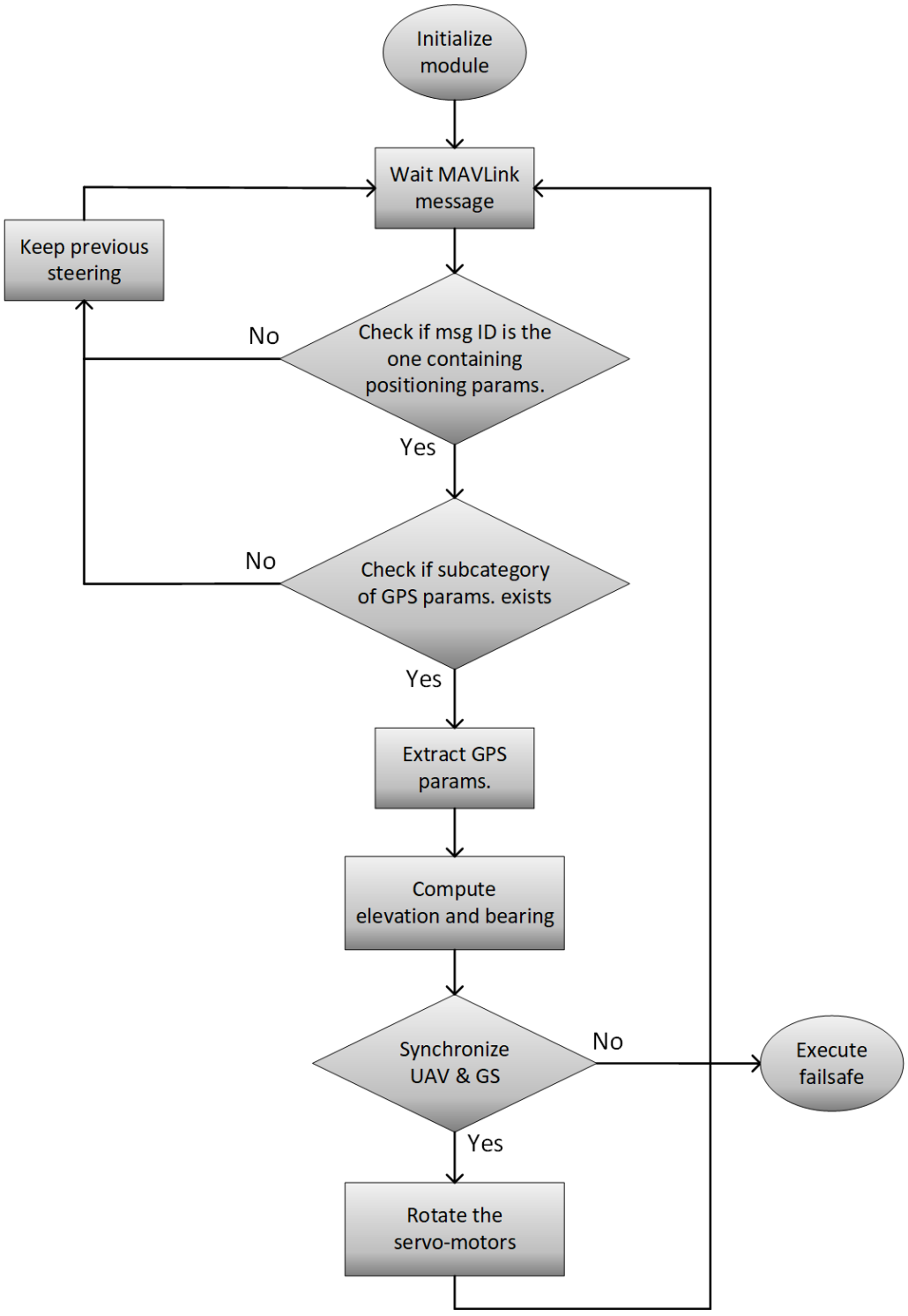


Figure 3.1 *Steering loop to be performed sequentially providing the steering capabilities of the system.*

- $\phi_2 \rightarrow$ Latitude of the ground station antenna.
- $\lambda_2 \rightarrow$ Longitude of the ground station antenna.
- $l \rightarrow$ Relative altitude of the UAV.
- $h \rightarrow$ Total height or altitude of the ground station antenna.

- Output parameters:
  - $\xi_1 \rightarrow$ Elevation angle for the ground station antenna.
  - $\theta_1 \rightarrow$ Bearing angle for the ground station antenna with respect to the North.
  - $\xi_2 \rightarrow$ Elevation angle of on-board antenna of the UAV.
  - $\theta_2 \rightarrow$ Bearing angle of on-board antenna of the UAV with respect to the North.

In Fig. 3.2 and Fig. 3.3 it is depicted a diagram showing the general scheme of the scenario to be develop during the design process of the algorithm.

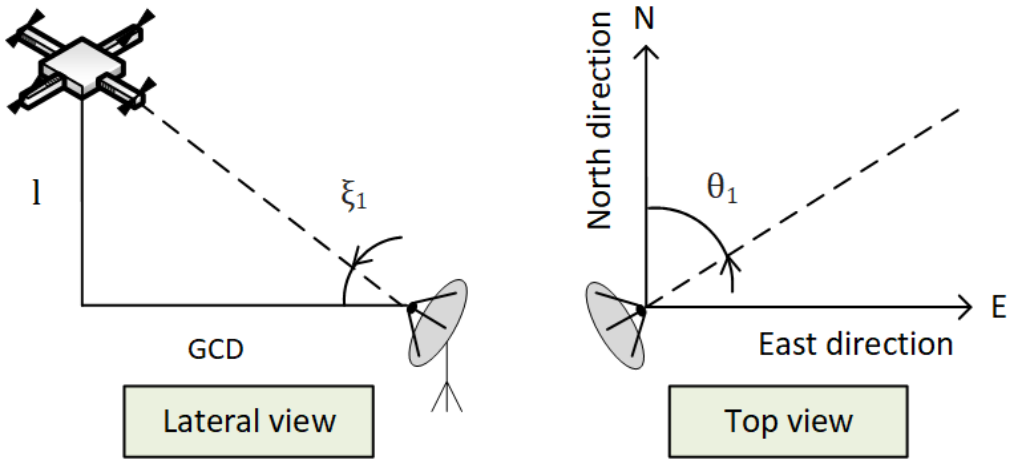


Figure 3.2 *Scheme showing the main parameters playing a role in the computations regarding the ground station antenna's end.*

These are the angles needed to point the antennas to each other. In the following lines, it is presented the procedure to obtain these final parameters, performed with Python programming language.

At this point, it is important to mention that as we are considering small-medium distances for the project (maximum 2 [km]), a model considering the Earth as flat is taken, as the one studied in [23]. This approximation simplifies the equations by

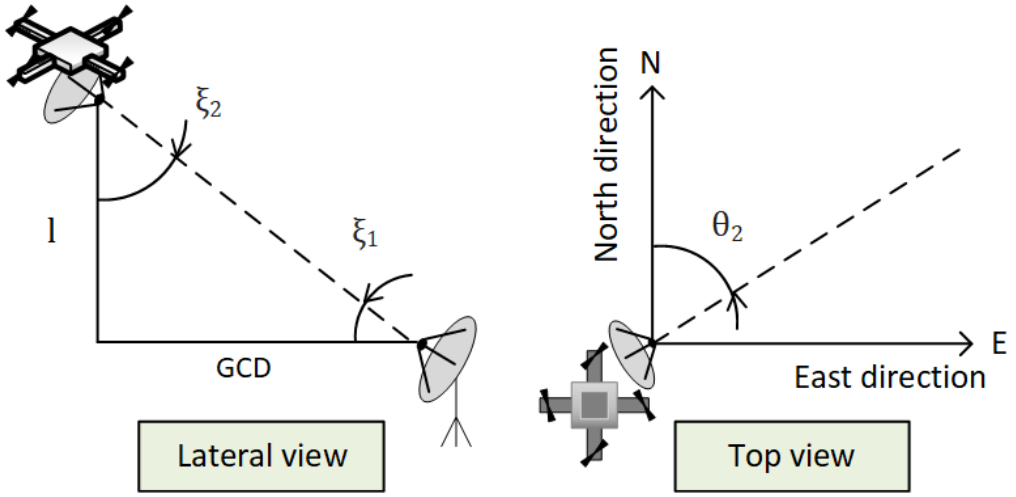


Figure 3.3 *Scheme showing the main parameters playing a role in the computations regarding the UAV's end.*

adding the prize of introducing an error. However, this error is small enough to be rejected at this distance.

The curvature of the Earth can be observed from the distance shown in Equation 3.1[1].

$$d \approx 3.57 \cdot \sqrt{h}, \tag{3.1}$$

where d is the distance of the horizon line in kilometers and h is the height of the object in meters.

Supposing a height $h = 4\,[m]$, the distance to the horizon reads $d = 7.14\,[km]$. Considering that our prototype is flying at much lower distances, the error is negligible by using this approximation.

For future works, if the tracking system was used for high-distances pointing, the model should be rethought to include this error.

3.1.1 Computation of the elevation angle for the ground station antenna

In this subsection, it is obtained the elevation angle to point the ground station antenna. In Fig. 3.4 a diagram is depicted to make the problem clearer.

[1]See "Wikipedia, distance to the horizon", 2017: https://en.wikipedia.org/wiki/Horizon

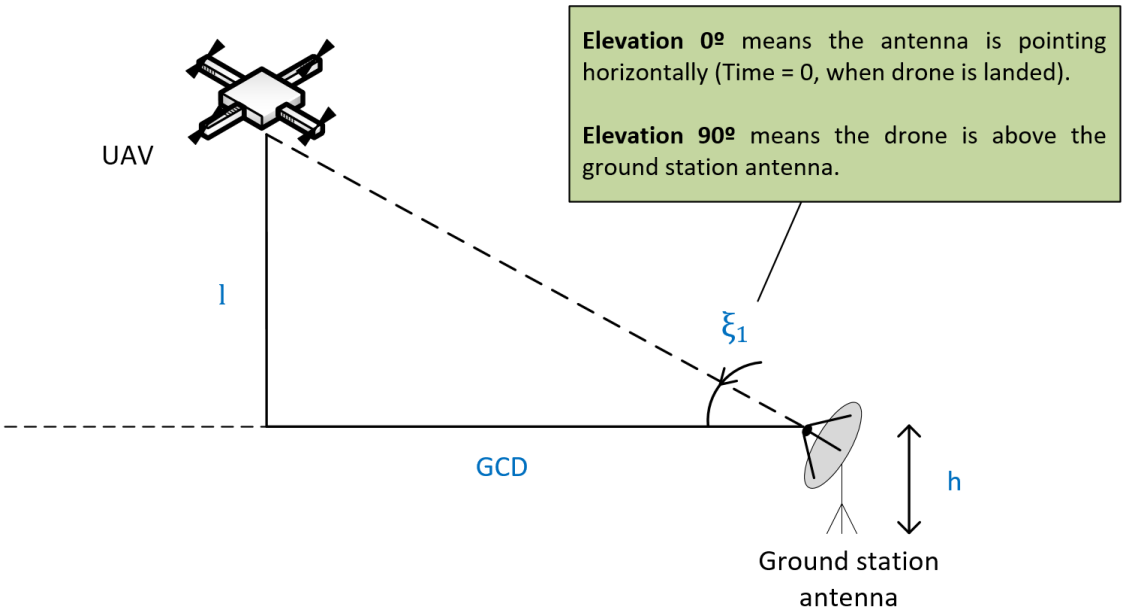


Figure 3.4 *Diagram showing the parameters involved in obtaining the elevation angle of the ground station antenna, represented as* ξ_1.

By looking at the diagram shown above, one realizes that four main parameters are needed to compute the elevation:

- GPS position of the UAV: Latitude + longitude in radians.
- Relative altitude of the UAV: Altitude with respect to the ground.
- GPS position of the ground station antenna: Latitude + longitude in radians.
- Relative altitude of the ground station antenna: Height of the tripod.

Note that these parameters stated above need to be known in both ends of the system at the same time in order to compute the angle of elevation. As a result, it will be necessary to establish a communication between UAV and ground station antenna to interchange these variables and make them available in both ends. This link is what we know under the name of telemetry link.

Once understood this, we can start making the mathematical calculations need to obtain the elevation angle.

The angle ξ_1 is obtained from Equation 3.2 and 3.3:

$$tan(\xi_1) = \frac{l - \sum_{i=1}^{n} h_i}{GCD}, \tag{3.2}$$

and therefore,

$$\xi_1 = arctan(\frac{l - \sum_{i=1}^{n} h_i}{GCD}), \tag{3.3}$$

where l is the altitude of the UAV, obtained from MAVLink messages and GCD represents the great circle distance between the UAV and GS. This parameter represents the shortest distance over the Earth's surface.

This great-circle distance can be computed by using the Haversine formula in Equation 3.4, as explained by the authors in [24]:

$$d = R \cdot c, \tag{3.4}$$

where R refers to the Radius of the Earth $(6,341\,[km])$ and c and a are computed as in Equation 3.5:

$$\begin{aligned} c &= 2 \cdot atan2(\sqrt{a}, \sqrt{1-a}), \\ a &= sen^2(\frac{\phi_2 - \phi_1}{2}) + cos(\phi_1) \cdot cos(\phi_2) \cdot sen^2(\frac{\lambda_2 - \lambda_1}{2}), \end{aligned} \tag{3.5}$$

where the function $atan2(\cdot)$ uses the signs of both arguments to calculate the quadrant of the result. It is defined as in Equation 3.6:

$$atan2(x,y) = \begin{cases} \arctan(\frac{y}{x}) & \text{if } x > 0 \\ \arctan(\frac{y}{x}) + \pi & \text{if } x < 0 \; and \; y \geq 0 \\ \arctan(\frac{y}{x}) - \pi & \text{if } x < 0 \; and \; y < 0 \\ +\frac{\pi}{2} & \text{if } x = 0 \; and \; y > 0 \\ -\frac{\pi}{2} & \text{if } x = 0 \; and \; y < 0 \\ Undefined & \text{if } x = 0 \; and \; y = 0 \end{cases} \tag{3.6}$$

With this procedure, the program will obtain the value in radians corresponding to the elevation of the ground station antenna.

3.1.2 Computation of the elevation angle for the on-board antenna of the UAV

In this section, the elevation angle to apply to the on-board antenna integrated in the UAV is computed. Fig. 3.5 showns a diagram with the general squeme.

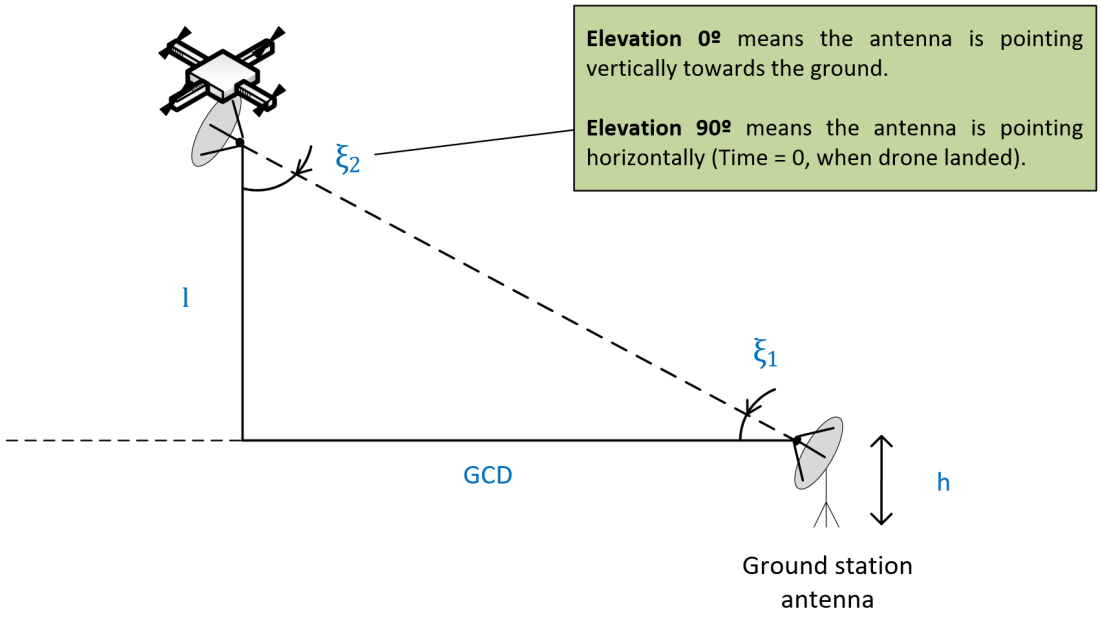


Figure 3.5 *Diagram showing the parameters involved in obtaining the elevation angle of the on-board antenna of the UAV, represented as* ξ_2.

By looking at the diagram, we can see some similarities with the previous section. For the obtaining of this new parameter, we are working with the same values considered before. The difference is that now we have already obtained the elevation angle of the ground station antenna, which can be reused to compute the elevation angle of the antenna in the UAV.

And so, the only parameter needed to compute the elevation of the on-board antenna of the UAV is the elevation angle of the ground station antenna, computed before.

According to the diagram showed above, the elevation angle for the UAV's antenna can be directly computed as $\xi_2 = 90 - \xi_1$. By reusing the elevation angle of the ground station antenna, some heavy and repetitive calculations can be avoided, and the program can be executed faster.

3.1.3 Computation of the bearing angle for the ground station antenna

Before performing the calculations to obtain θ_1, let's first explain in detail what the bearing is.

The bearing angle measures the acute angle a path or line of sight (LOS) makes with a fixed North-South line. This bearing angle can be measured from 0 to 180° with respect to the cardinal points.

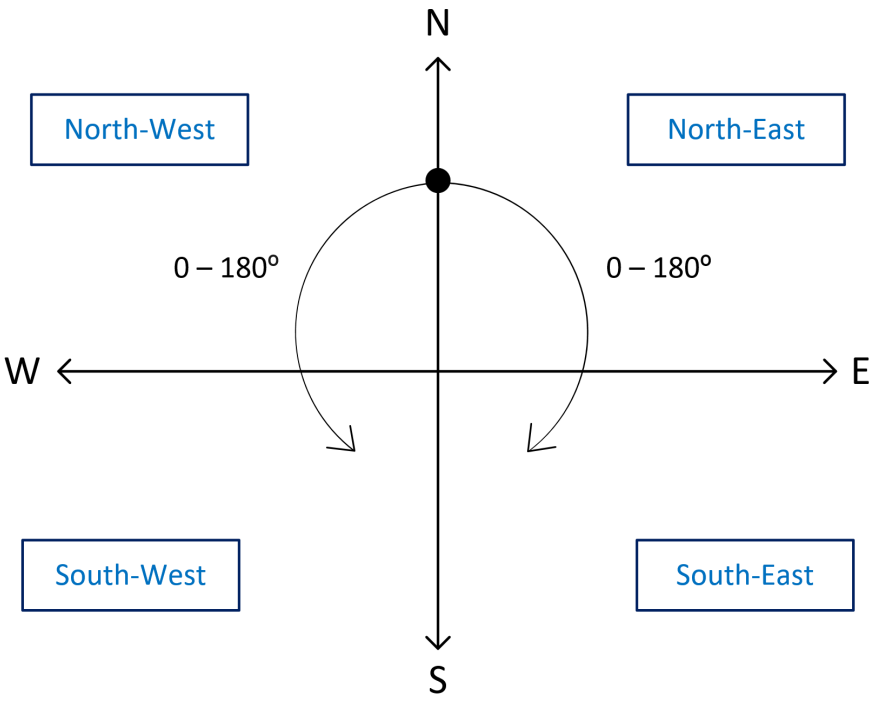


Figure 3.6 *Formal definition of bearing angle θ_1 with respect to the North cardinal point.*

In a general context, the bearing angle shows the information of the direction to 'start walking' to reach the destination coordinates by following the shortest path. For our purpose, this is translated to where to point the antenna in the direction of the destination coordinates (coordinates of the UAV or the ground station, in each case).

Once this is understood, the mathematical procedures to its calculation are shown. The bearing angle θ_1 for the ground station antenna expressed in latitude [radians] + longitude [radians] can be computed as shown in Equation 3.7:

$$\theta_1 = atan2(sin(\lambda_2 - \lambda_1) \cdot cos(\phi_2), cos(\phi_1) \cdot sin(\phi_2) - sin(\phi_1) \cdot cos(\phi_2) \cdot cos(\lambda_2 - \lambda_1)), \tag{3.7}$$

where ϕ_1 corresponds to the latitude of the UAV, ϕ_2 represents the latitude of the GSA, λ_1 refers to the longitude of the UAV, and λ_2 corresponds to the longitude of the GSA. All these parameters are expressed in radians.

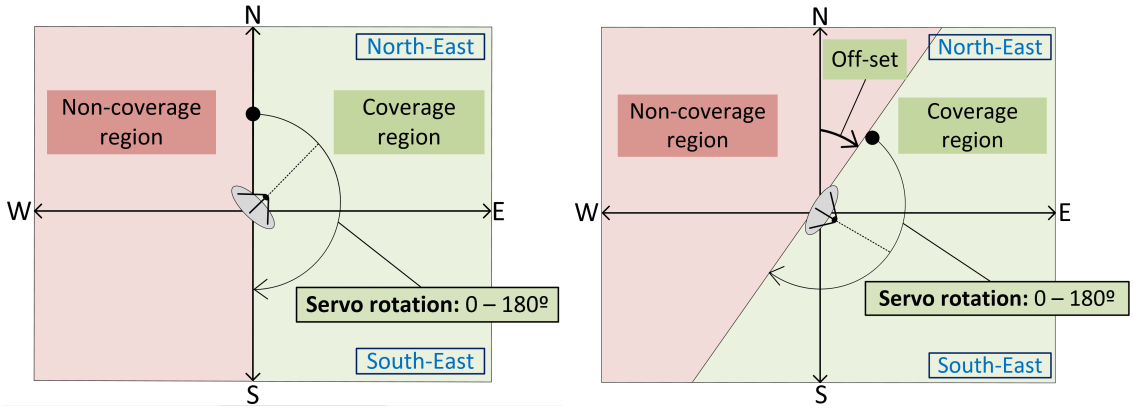
The equation 3.7 outputs a numerical value between $-180°$ and $180°$. For our purpose, it is easier to have a value between $0°$ and $360°$, so we just need to add $360°$ if the value obtained is negative.

The value previously computed represents the angle to rotate the antenna with respect to the North cardinal point. By having a compass in the ground station antenna, we can point it to the magnetic North and apply directly the angles obtained.

However, it is important to mention that the servo motors to be used only provide a rotation between $0°$ and $180°$. This has to be considered when configuring the mission plan of the UAV and when placing the ground station antenna.

With this angle of rotation for the servo motors, the ground station antenna can only cover a region of $180°$, so it should be placed pointing carefully to the flight region in which the UAV is going to operate.

Two examples are shown in Fig 3.7 to make the problem clearer.



(a) Coverage region 1 (b) Coverage region 2

Figure 3.7 *Coverage regions considering 180° for the rotation of our servo motors.*

If the antenna is pointed in a different direction from the magnetic North, the

calculation of an additional off-set angle is necessary to point the antenna properly. Recall that the bearing angle is a value between 0° and 360°, always with respect to the North.

To set this off-set angle properly, the heading of the ground station antenna should be obtained by the compass module when the horizontal servo motor is rotated 0°. Then our angle to rotate the antenna will be the bearing minus this calculated off-set.

Currently in the market places, there are many models of servo motors with different specifications. Servo motors with rotation angles of 90°, 180°, 360°, or even with continuous rotation angles can be found with similar prizes. We could have considered different models providing 360° rotation to provide a complete coverage region around the antenna, but the problem comes at the time of wiring the mechanical rotation system. The pieces and mechanism necessary to provide a complete rotation without twisting the cables are very expensive. For now, as we are building a first prototype for the project, we will avoid buying this expensive equipment.

3.1.4 Computation of the bearing angle for the on-board antenna of the UAV

The calculations performed to compute θ_2 are similar to the ones presented in the previous section. For the on-board drone's antenna, we have the same scenario that the one presented in Section 3.6.

Therefore, the numerical value corresponding to θ_2 can be done with the same expression, presented in Equation 3.8:

$$\theta_2 = atan2(sin(\lambda_1 - \lambda_2) \cdot cos(\phi_1), cos(\phi_2) \cdot sin(\phi_1) - sin(\phi_2) \cdot cos(\phi_1) \cdot cos(\lambda_1 - \lambda_2)), \tag{3.8}$$

where ϕ_1 corresponds to the latitude of the UAV, ϕ_2 represents the latitude of the GSA, λ_1 refers to the longitude of the UAV, and λ_2 corresponds to the longitude of the GSA. All these parameters are expressed in radians.

As exposed before, with the equation it is obtained an angle between $-180°$ and 180° that will be converted to an angle between 0° and 360° afterwards.

With this previous result, we will be able to point the on-board antenna in the

proper direction to establish successfully the communication link with the ground station antenna.

The fragment of Python code to perform the process described along this section is presented in Algorithm F.1.

3.1.5 Parameters needed to perform the computations

Once elaborated the methods and calculations to obtain the angles of elevation and bearing of both antennas, let's define a list of parameters needed for these previous calculations.

It is important to know well what variables we need, because they will have to be extracted from MAVLink messages being interchanged within MAVProxy (Section 3.2).

The list of parameters to be extracted in the next section are the ones presented in Table 3.1.

Table 3.1 *List of parameters needed for the computation of elevation and bearing of both antennas and where to extract them from.*

Needed parameter		**Where it is extracted from**
Elevation	GS antenna's latitude	GPS or manually
	GS antenna's longitude	GPS or manually
	UAV's latitude	MAVLink messages
	UAV's longitude	MAVLink messages
	UAV's altitude	MAVLink messages
Bearing	GS antenna's latitude	GPS or manually
	GS antenna's longitude	GPS or manually
	GS antenna's heading	Compass
	UAV's latitude	MAVLink messages
	UAV's longitude	MAVLink messages
	UAV's heading	MAVLink messages

The GPS coordinates of the ground station antenna can be obtained either using a GPS module. As the ground station antenna is placed to be static, these coordinates will not vary along time, and will be enough to obtain them manually.

3.2 MAVLink messages: Extraction of needed parameters

In Section 2.1.1 the basic structure of a MAVLink message is explained. A MAVLink message uses a sequence of codified bytes to send the information within the sys-

tem. There are twenty three types of MAVLink messages that can be transmitted, identified in the field 'Message ID'.

For our purpose, we are going to focus just in the 'MAVLINK_MSG_ID_PARAM_REQUEST_READ' message type. This message is transmitted periodically every one second.

This is the message type containing the complete list of parameters transmitted in real time within MAVLink. Depending on the version and type of UAV used, the number of parameters can vary. For a typicall quadcopter, the list contains 537 parameters.

Inside this list, the parameters are classified in subcategories with specific names indicating what is being transmitted within.

In the official webpage of MAVLink [17] it is shown the list with all the subcategories and parameters included in the messages, so we can search the list in which the ones desired are included.

According to the previous section, we need the following parameters to compute the elevation and bearing for both antennas:

- Coordinates: Current latitude and longitude of the UAV.
- Altitude: Altitude of the UAV.
- Heading: Angle the UAV is pointing at.

The subcategory 'GLOBAL_POSITION_INT' is the one containing these parameters, among others. The complete list of parameters contained here is shown in Table 3.2.

Once known in which kind of message ID and in which category the parameters are included, we can program an algorithm to extract them. The flow of the algorithm is shown in Fig. 3.8.

This program will extract the values of 'lat', 'lon', 'alt' and 'hdg', necessary for the computations of the final angles elevation and bearing.

The complete algorithm that manages the extraction of these parameters is shown in Algorithm F.2. The code is executed within MAVProxy, in which all MAVLink messages are being transmitted during the flight.

Table 3.2 *List of parameters, with their type and description, contained in the subcategory 'GLOBAL_POSITION_INT'.*

Field name	Type	Description
time_boot_ms	uint32_t	Timestamp
lat	int32_t	Latitude in degrees $\cdot 10^7$
lon	int32_t	Longitude in degrees $\cdot 10^7$
alt	int32_t	Altitude in meters
relative_alt	int32_t	Altitude above ground in meters
vx	int16_t	Ground speed in X axis expressed in [m/s] $\cdot 100$
vy	int16_t	Ground speed in Y axis expressed in [m/s] $\cdot 100$
vz	int16_t	Ground speed in Z axis expressed in [m/s] $\cdot 100$
hdg	uint16_t	Heading on the UAV in degrees $\cdot 100$. Range is 0 - 359°

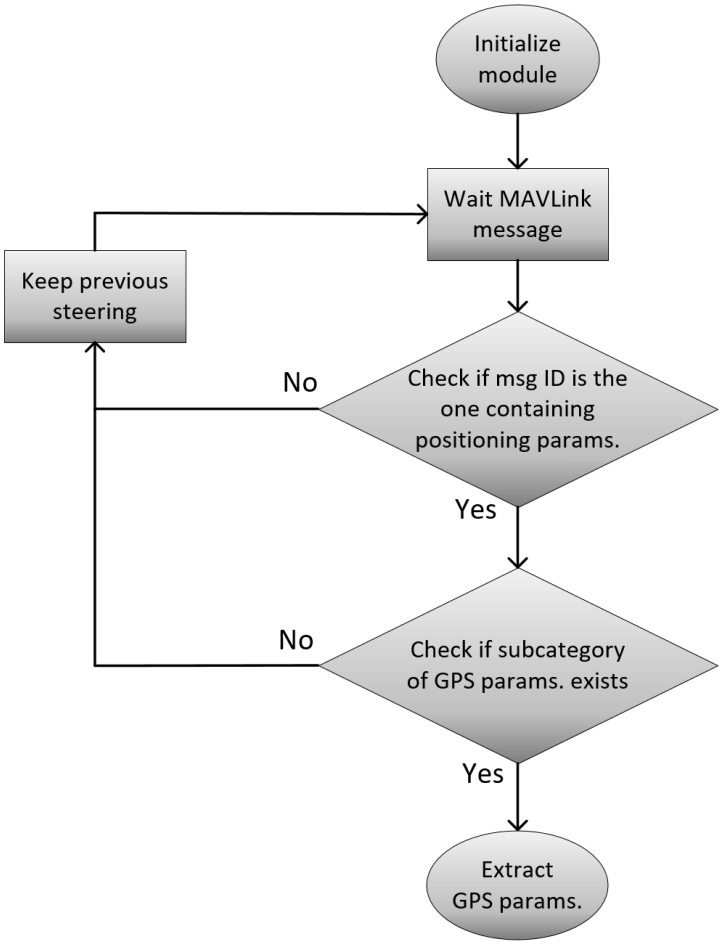


Figure 3.8 *Flow of the algorithm to be performed in the BeagleBone to obtain the desired parameters from the MAVLink messages.*

As a consequence of the extraction of this values, the program is ready to compute the final angles of elevation and bearing, but do not forget that a synchronization step to share these values is required between the UAV and the ground station. The way of transmitting the values and computing the variables is explained in the next section.

3.3 Ground station and UAV real time synchronization

As it has been introduced in previous sections, a synchronization process between the UAV and the ground station is key to make the whole system work properly. This communication is exchanging parameters and variables between both ends in order to make the proper calculations for the elevation and bearing to apply to both antennas.

Previous sections explain the functions and methods used to compute the final angles of elevation and bearing. In contrast, this section explains where this computations are done, and how they are transmitted and synchronized within the system.

Recall that in Section 3.1 the parameters needed to make the calculations are introduced. This means that we need the complete set of these parameters when computing the final angles of elevation and bearing. At this point, we can make two assumptions:

- The ground station antenna is considered to be static.
- The UAV is considered to be moving.

As a consequence of this assumption, the GPS coordinates and heading of the ground station antenna can be obtained and given to the UAV before the starting of the program, so it will not be necessary to share them periodically with the UAV, as they are not varying.

At this point, and recalling again Section 3.1, we can see that if the parameters regarding the ground station antenna ('GS antenna's latitude', 'GS antenna's longitude' and 'GS antenna's heading') are considered static and already known by the drone before starting the program, we can already make the computations in the UAV's end to point both antennas, because all variables are known in the UAV (drone's parameters are already extracted from MAVLink packets).

At this point, we have to possibilities to compute and synchronize the final values:

- Option 1: Transmit UAV's parameters ('UAV's latitude', 'UAV's longitude', 'UAV's altitude' and 'UAV's heading') to the ground station, make the mathematical calculations in the ground station and transmit the computed angles of elevation and bearing corresponding to UAV's antenna back.
- Option 2: Make the mathematical calculations in the UAV's end and transmit the angles of elevation and bearing corresponding to ground station antenna to the ground station.

One can guess the simplicity of the second option, avoiding an extra communication process, which takes extra time and resources. It is because of that the second option is the one adopted for this project.

Summing up, as the parameters regarding the ground station are static, all the computations are done in the on-board computer of the drone, the BeagleBone. The elevation and bearing for the UAV's antenna will be directly applied, while the elevation and bearing for the ground station antenna are send to the ground station, which will apply them to its antenna.

Fig. 3.9 depicts a general scheme of the communication process followed by the system.

Once understood how the synchronization process is made and where it is actually been executed, let's explain how the variables and values are transmitted within both ends.

The communication is made by sending UDP datagrams through the backbone link. Basically, the UAV is sending UDP packets during a configurable interval of time to the static IP address of the ground station, and the ground station is receiving these packets at the same time.

The packets sent through the link contains the final angles of elevation and bearing for the ground station antenna. Before these variables are sent, they are serialized by using a tool as pickle or json. In the ground station end, the data is deserialized and then extracted to be processed.

Once the timeout is triggered, the sender (UAV) stops sending packets, and the mechanical structures in the UAV and ground station start to steer the antennas to the proper angles. This process is repeated as a loop to keep steering the antennas during all flight.

The algorithm implemented in the UAV, acting as sender, to transmit the UDP

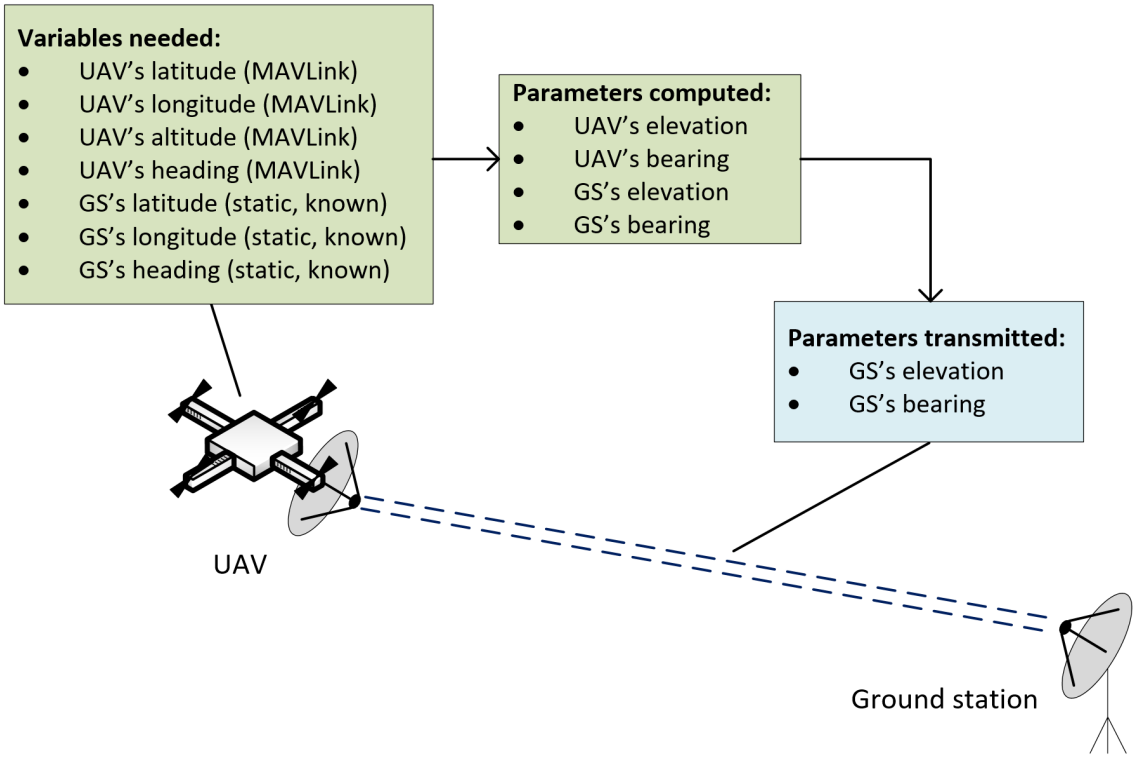


Figure 3.9 *Synchronization process between UAV and GSA. Elevation and bearing angles of the GSA are transmitted to the ground station.*

packets is presented in Algorithm F.3.

The algorithm implemented in the ground station, acting as receiver, just manages the reception of the UDP datagrams and the deserialization to convert them into readable data.

As a result, once received the messages from the UAV, the variables containing the final angles will be available in the ground station, ready to be used by the computer and servo motors to point both antennas.

At this point, with all the parameters obtained and synchronized, it is only left to configure the servo motors to rotate the antennas to the desired angles. Remember we have two servo motors for each antenna, one of them rotating to provide the elevation, and the other rotating to provide the bearing or azimuth. In the next section is presented how to configure the servo motors to make this possible.

3.4 Servo motors configuration

Once computed and synchronized the key angles of elevation and bearing for both antennas, we can start configuring the servo motors to provide the desired rotation.

First of all, let's choose the models of the servo motors to be integrated with each antenna. To do so, we need to study certain specification of the servo motors and make sure they fit well with our application. The election of the servo motors for each antenna is done in the next section, according to key features such as torque or weight.

3.4.1 Servo motors for the on-board antenna of the UAV

Let's choose the servo motors to be mounted in the UAV's on-board antenna to provide the automatic steering. This antenna has a weight of 155 [g] approximately, so a servo motor with a torque greater than this weight should move it properly.

The weight of the servo motors are also important, as they are going to be mounted in the target UAV. Remember the less weight we put in the UAV, the better flight times it will get. An acceptable weight for the servo motors could be < 20 [g] for each one.

The rotation angle chosen for the servo motors is from 0° to 180°. The rotation speed of the servo motors is high enough for our application.

The input voltage for the servo motors is between 3 and 6 [V], so they can be feed directly from the BeagleBone or from an adjustable voltage regulator connected to the battery (See Appendix C).

This input voltage range means that with the maximum volts applied in its input, the servo motor will provide the maximum torque available.

By having this in mind, the final brand and model chosen for the servo motors is: HEXTRONIX HXT-900[2].

The main specifications of the servo motors concerning our project are shown in Table 3.3.

As we can see, the torque the servo motor is capable to provide in its rotation axis is much greater than the weight of the antenna, so it should move the antenna easily.

The weight of the combined two servo motors is 18 [g], acceptable for our purpose.

[2]See "Servo motor HEXTRONIX HXT-900 on-line specifications", 2016: https://hobbyking.com/es_es/hxt900-micro-servo-1-6kg-0-12sec-9g.html

Table 3.3 *Specifications of the servo motor HEXTRONIX HXT-900.*

Parameter	Value
Torque	1.6 [Kg-cm] when 6 [V] applied
Speed	0.12 [sec/60°]
Rotation angle	0 - 180 °
Weight	9 [g]
Input voltage	3 - 6 [V]

3.4.2 Servo motors for the ground station antenna

Let's choose now the brand and model of the servo motors for the ground station antenna.

The ground station antenna weights approximately 400 [g]. The mounted structure to hold it still and the rotatory base weights approximately 150 [g]. Therefore, the total weight of the ground antenna system can be approximated to 600 [g]. For the selection of the servo motor, we have to reject options with a torque lower than this value.

For this scenario, the weight of the servo motors is not important, as the base station is going to be placed static in the ground. Although this parameter is presented, it is not considered at the time of selecting the model.

The rotation angle chosen for the servo motors is the same as before, from 0° to 180°. The rotation speed of the servo motors is high enough for our application.

The input voltage range to apply to the servo motors chosen[3] is from 6 to 7 [V], so we need a regulator connected to the battery to provide this amount of voltages.

The specifications of this servo-motor are shown in Table 3.4.

3.4.3 Servo motors operation and configuration

Once bought the models to be used for the servo motors, we can start configuring them to provide the desired angles for the tracking system. The servo motors are going to be configured to work with the Beable-Bone/Raspberry Pi.

These boards have several general port input output (GPIO) ports for controlling

[3]See "CORONA DS339-HV web specifications", 2017: https://hobbyking.com/en_us/corona-ds339hv-digital-metal-gear-servo-5-1kg-0-13-sec-32g.html

Table 3.4 *Specifications of the servo motor CORONA DS339-HV.*

Parameter	Value
Torque	5.1 [Kg-cm] when 7.2 [V] applied
Speed	0.14 [sec/60°]
Rotation angle	0 - 180 [°]
Weight	32 [g]
Input voltage	6 - 7.2 [V]
Input current	420 [mA]

external devices such as the servo motors. The configuration and operation is shown in the following section.

3.4.4 Control signal

The servo motors need a control signal to set the rotation angle to be spun. This signal is a 50 [Hz] (puses sent every $T = 20$ [ms]) pulse width modulation (PWM) signal with a variable duty cycle between 5 and 10 % to control the position of the servo motor, as presented in Fig. 3.10.

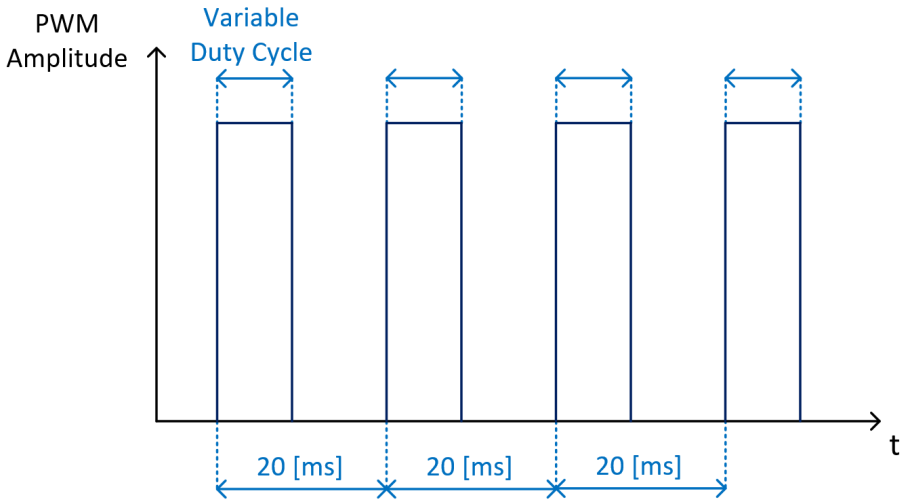


Figure 3.10 *PWM signal to apply to the servo motors to control their angle of rotation. The frequency of the signal is 50 [Hz] with a variable duty cycle.*

The desired PWM signal can be obtained from the BeagleBone or the Raspberry Pi by using the library 'Adafruit.BBIO.PWM'. This library provides all the necessary methods and functions to manipulate the PWM signals in the output ports.

A simple initialization for a PWM signal can be done as shown in Algorithm 1.

Algorithm 1 Initialization of a PWM signal in a Beable-Bone or Raspberry Pi.

Ensure: 50 [Hz] PWM signal with 5% duty cycle in GPIO no. 16

```
1: import Adafruit_BBIO.PWM as PWM
2: servoPinV = "P9_16"
3: initialDCV = 5
4: PWM.start(servoPinV, initialDCV, 50)
```

Once it is known how to configure and manipulate the servo motors, let's calibrate both models in order to use them in the final system.

3.4.5 Calibration

The servo motors should be calibrated before their final use. Calibration means to obtain the value of the duty cycle to rotate the servo motor fully right (0°) and the value of the duty cycle to rotate the servo fully left (180°).

For the servo motors mounted in the on-board antenna, the following parameters are obtained after the calibration:

- Fully right or 0 °: Duty cycle = 2.6 %.
- Middle position or 90 °: Duty cycle = 7,45 %.
- Fully left 180 °: Duty cycle = 12.3 %.

The position of the servo motor is set by the length of a pulse received every 20 [ms] ($f = 50$ [Hz]). According to the calibration process made above,if that pulse is high for 0.52 [ms], the servo motor will be rotated to 0 ° (fully left); if it is 1.52 [ms], it will be rotated to its medium position; and if it is 2.52 [ms], it will be rotated to 180 °. Fig. 3.11 illustrates this process.

The duty cycle of a PWM signal can be obtained with Equation 3.9:

$$DC\,[\%] = \frac{PW}{T} \cdot 100, \tag{3.9}$$

where DC represents the duty cycle, PW indicates the pulse width and T corresponds to the period of the signal, being $T = 20\,[ms]$.

The servo motors for the ground station antenna are calibrated the same way done before. The procedure done is almost the same, so it is omitted. The final results of the calibration for these servo motors are:

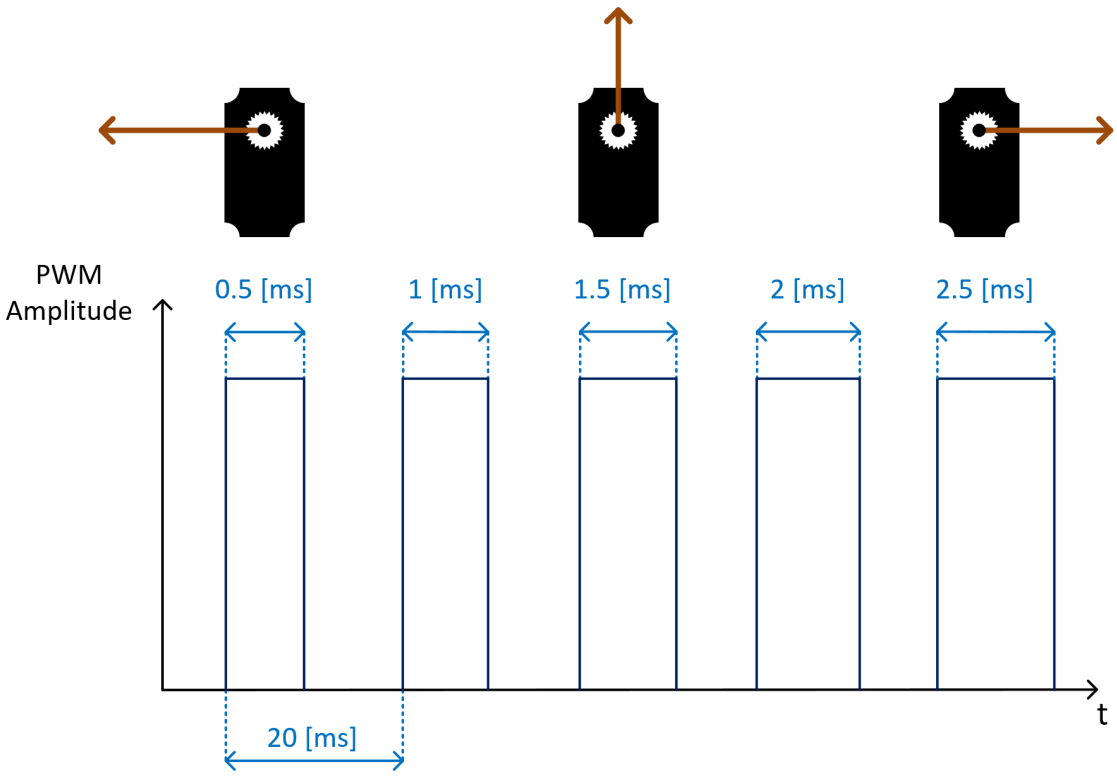


Figure 3.11 *Calibration process of the servo motor. Obtaining of the values of the duty cycle to rotate the servo motor fully right and fully left.*

- Fully right or 0 °: Duty cycle = 2.3 %.
- Middle position or 90 °: Duty cycle = 7,15 %.
- Fully left or 180 °: Duty cycle = 12 %.

And therefore, the fully right position (0 °), middle position (90 °) and fully left position (180 °) are obtained with pulses with a length of 0.46 [ms], 1.43 [ms] and 2.4 [ms], respectively.

3.4.6 From angles to duty cycle

In the previous calibrations, three connections between angles and duty cycles are obtained. We know the duty cycles to apply to obtain 0 °, 90 °, 180 °, but not the duty cycles to apply for other angles.

As it is much simpler, we would like to specify the angle instead of the duty cycle to rotate the servo motor to the desired position, so let's make a linear equation that relates the duty cycle with to the angle of rotation.

As this equation will be a linear function, we can use these points obtained above to compute and characterize it.

Generally:

$$\begin{aligned} DC_1 &\Longleftrightarrow 0\,^\circ, \\ DC_2 &\Longleftrightarrow 90\,^\circ, \\ DC_3 &\Longleftrightarrow 180\,^\circ, \end{aligned}$$

where $DC_n,\ n = 1, 2, 3$ corresponds to the applied or desired duty cycle to obtain these angles.

The previous connections can be taken to the real plane as points:

$$\begin{aligned} &(0, DC_1), \\ &(90, DC_2), \\ &(180, DC_3). \end{aligned}$$

An straight line can be traced from the first point to the third one. The slope of this line is as in Equation 3.10.

$$m = \frac{DC_3 - DC_1}{180 - 0} = \frac{DC_3 - DC_1}{180}. \tag{3.10}$$

And finally, the equation relating the desired angle, represented as α, and duty cycle is obtained as in Equation 3.11 and 3.12:

$$\begin{aligned} DC - DC_1 &= m \cdot (\alpha - 0), \\ DC - DC_1 &= \frac{DC_3 - DC_1}{180} \cdot (\alpha - 0), \end{aligned} \tag{3.11}$$

therefore:

$$\begin{aligned} DC - DC_1 &= m \cdot (\alpha - 0), \\ DC - DC_1 &= \frac{DC_3 - DC_1}{180} \cdot (\alpha - 0), \\ DC &= \frac{DC_3 - DC_1}{180} \cdot \alpha - DC_1. \end{aligned} \tag{3.12}$$

Now, replacing the values obtained for our specific models of servo motors, it is obtained:

- Servo motors for the on-board UAV's antenna will follow the Equation 3.13:

$$DC = \frac{12.3 - 2.6}{180} \cdot \alpha - 2.6 \rightarrow DC = \frac{9.7}{180} \cdot \alpha - 2.6. \tag{3.13}$$

- Servo motors for the ground station antenna will follow the Equation 3.14:

$$DC = \frac{12 - 2.3}{180} \cdot \alpha - 2.3 \rightarrow DC = \frac{9.7}{180} \cdot \alpha - 2.3. \tag{3.14}$$

With this equation, we can convert easily from angles to duty cycles and the other way round. So we can work with angles instead of duty cycle, which is easier.

Now that we have created this conversion and know how to manipulate the servo motors to provide a specific rotation angle, we can program our algorithm that will handle its movement. The code to perform that action is shown in Algorithm F.4.

It is important to note the sentence *time.sleep(0.1)* in the code fragment. This command waits 0.1 [s] before moving a degree in each elevation and azimuth angle. This will slow the speed at which the servo motor is moving. It is better to move the servo motors smoothly in order to not distort the stabilization of the UAV. This rotation speed can be changed anytime.

In this section, the movement of the servo motors to point the antennas has been explained. Now, it is only left the integration of all of these functions and calculations within MAVProxy.

To make it possible, a new module integrated with MAVProxy is created and configured to operate once necessary. This process is explained in the next section.

3.5 MAVProxy module: algorithm unification

At this point of the document, all the independent functions to point both antennas in the right direction have been detailed. It has been presented the method to compute the angles of elevation and bearing for both antennas, how to perform the synchronization process between UAV and ground station and finally the function to move accordingly the servo motors.

In this section, all these functions and methods are integrated to provide the proper autonomous steering system. The result can be seen as a sequential program that executes these functions accordingly to provide the final goal.

This program is going to be run in both UAV and ground station ends to provide the steering mechanism in both antennas.

3.5.1 MAVProxy integrated module

When talking about the UAV, the best way to run our own steering algorithm is to integrate this software with the kernel of MAVProxy. MAVProxy has several installed-by-default modules to perform certain basic actions (e.g. take-off, read battery percentage, change flight mode, etc.).

In this document it is proposed to create a completely new module and integrate it with MAVProxy. This new module would contain all the necessary algorithms to compute the angles to steer both antennas, make the communication between UAV and GS, and mechanically rotate the antenna of the UAV. Remember we are at the UAV's end, only steering the UAV's antenna at this point. The ground station will steer its antenna according to the parameters received from the UAV.

The integration of our module with the kernel of MAVProxy gives us great advantages for this particular application:

- Utilization of useful functions programmed by default in MAVProxy.
- Acces to the complete list of parameters being transmitted periodically within MAVLink messages.

This second point is key for our project, as it was explained in Section 3.2, it is essential to extract certain parameters in order to compute the angles to point the antennas. By having our module integrated with MAVProxy, we can easily access the list of parameters to search for the ones required.

Once understood the advantages of this, let's create our proper module and its integration within MAVProxy.

The first thing to create within the module is the definition of its class, containing its attributes and initial values. The code corresponding to the class of the module is presented in Program F.5.

Note the sentence add_comand(...) in line 6. This order will add our module to the list of executable modules when running MAVProxy in the UAV. The module will appear now in the screen when asking for the available modules, and we will be able to start it from there whenever the steering system is required.

The first function to program in this module is the one that handles the extraction of the parameters within MAVLink messages. This function is presented in Algorithm F.2.

The next step is to build a function that is executed periodically and will make use of all methods and algorithms programmed to reach the final goal of the steering system. This is the key function to perform the steering process in our system. The iterative method is presented in Algorithm 2.

Algorithm 2 Iterative steering task to be executed sequentially in MAVProxy.

Ensure: One steering step for the steering system
{"'Function called periodically by mavproxy"'}
1: **while** Steering feature is required **do**
2: Usage of the function **'antennaTracking(...)'** to compute all the necessary angles to point both antennas
3: Usage of the function **'sendData(...)'** to build a list, serialize and transmit the parameters to the IP of the ground station
4: Usage of the function **'moveServos(...)'** to mechanically rotate both antennas and point them in the desired direction
{One iteration is complete}
5: **end while**

Note that the functions used in Algorithm 2 are the ones described within the chapter:

- Line 6: antennaTracking(...): See Algorithm F.1.
- Line 20: sendData(...): See Algorithm F.3.
- Line 26: moveServos(...): See Algorithm F.4.

In the ground station end, an algorithm handling the reception and movement of the servo motors is running at the same time. This code is executed in a loop to perform the steering task. The code is presented in Program F.6.

With this lines, the code will automatically receive the UDP datagrams coming from the UAV with the information of the angles to point the ground station antenna.

When these angles are received and processed, they will be given to the servo motors, which will move the antenna accordingly.

With the execution of these programs in both UAV and GS ends, the steering system is completed. This tasks will be executed in a loop (See Fig. 3.1) to keep pointing the steering system until the program is stopped.

In the following section it is presented the potential sources of error that can cause inaccuracies in the tracking system, leading to mispointing in the antennas mounted in the UAV and ground station.

3.6 Sources of error

Once implemented the autonomous steering algorithm with servo motor rotation for the antennas, the potential sources of errors that could affect the accuracy of the pointing technique are analyzed. This study can be taken as a basis to perform future improvements to the system.

It is important to understand that for this research thesis the system is desgined to work at short-medium distances (1 - 2 [km]), so some of the errors described in the section have a minimum effect on the link, but if we operated the link in high distances, they would affect drastically the accuracy of the pointing system.

The main sources of errors affecting the accuracy of the tracking system, sorted according to its relevance, are:

- Geodesic calculations: When the system is working in small-medium distances, a Flat-Earth model can be considered to calculate the elevation angle needed to be applied in both antennas. This approximation will lead to an error in the calculation of this angle. However, this miscalculation is negligible when operating the UAV nearby the ground station antenna.

  This error becomes non-negligible if the system is to be operated in high distances. To minimize the error, the calculations should be made according to a geodesic model that takes into account the curvature of the Earth.

- Inaccuracies in GPS: As the authors explain in their project work [25], GPS modules are susceptible to fail or give inaccurate results due to surrounding obstacles, such as mountains, vegetation or human made structures.

  The United States Government estimates 4 RMS (7.8 [m] with a confident

interval of 95%) for horizontal accuracy considering civilian GPS[4].

- Inaccuracies in compass: Similar to the previous point, compasses modules can also have innacuracies, affecting to the final results provided be the device [26]. The system works with two compasses, one in the UAV and the other in the ground station antenna. This could affect the precision of the system in the same way as before, leading to mispointings in both antennas.

  Compasses could provide a maximum error of $\pm$ 2°, according to the model selected [5].

- Inaccuracies in altitude: The UAV is the one managing the information regarding its altitude. To do so, most of commercial UAV use a barometric sensor to read the pressure of the air constantly. Before taking off, it registers the pressure of the air and stores this variable for future comparison. While flying, the value is compared to the previous to estimate the relative altitude. This estimation can lead to errors of a 1 - 2 [m] when calculating the relative altitude of the UAV with respect to the ground [27].

- Servo motor and mechanical structure errors: The inaccuracies in the proper structure to track the antennas are also playing its role in the steering system. Depending on the precision of the motors rotating the structure, the system will face smaller or bigger errors.

  It is important to realize that this error increases drastically with respect to distance. An error in 0.1° when working at 1 [km] can lead to a mispointing of approximately 2 [m], but the same error when working at 10 [km] can lead to a mispointing of 18 [m].

[4]See: "https://gis.stackexchange.com/questions/43617/what-is-the-maximum-theoretical-accuracy-of-gps"

[5]See: "https://sectionhiker.com/magnetic-compass-accuracy/"

4. VALIDATION OF THE SYSTEM

This chapter contains a validation process for the algorithm implemented. It is first shown the hardware configuration used along the project to make possible the steering algorithm. With this, it is included real pictures of the components playing an important role in the establishment of the backbone link.

Firstly, it is presented a theoretical simulation to prove that the software algorithm is working and outputting the expected values regarding elevation and bearing angle.

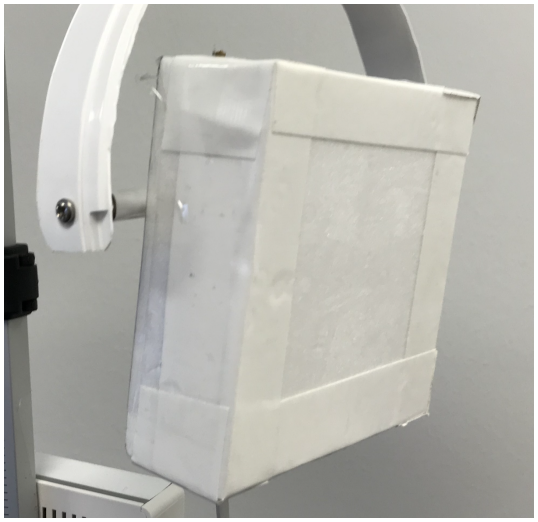
After this, five empirical tests have been carried out in a real scenario to characterize the behaviour of the algorithm. These tests compare the value of the received power using our algorithm with a reference value, allowing us to calculate an approximation of the error of the algorithm implemented during the research process.

4.1 Hardware and devices

The main components that make possible the establishment of the backbone link with the ground station are the on-board computer (BeagleBone), the integrated router, and the on-board antenna. These components are shown in Fig. 4.1.

(a) On-board computer or BeagleBone

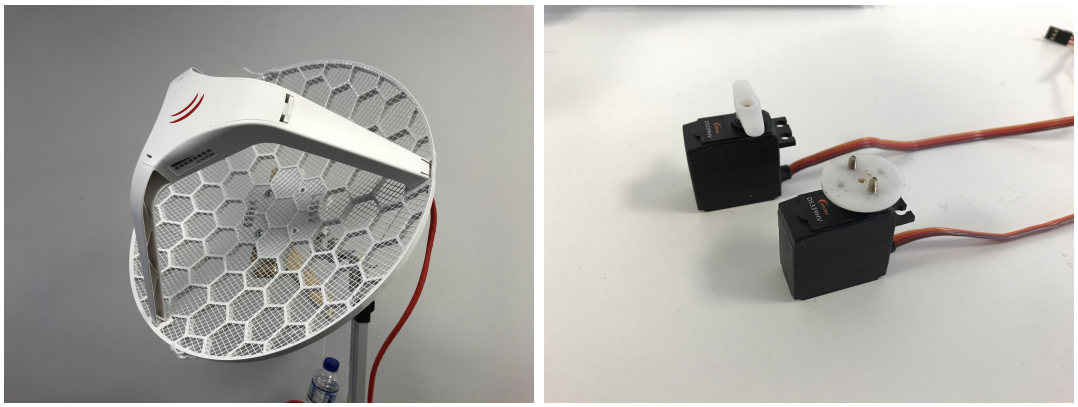
(b) On-board routing device



(c) On-board antenna

Figure 4.1 *Main equipment allocated in the UAV playing an important role at the time of establishing the backbone link between ground and air devices.*

In the other end, the main components regarding the ground station that allow to establish the backbone link in the opposite direction are the computer acting as the proper ground station and the ground station antenna. These devices are shown in Fig. 4.2. Additionally, the servo motors that have been configured during the thesis are shown in the same picture.



(a) Ground station antenna

(b) Servo motors to rotate the angles in the UAV's end

Figure 4.2 *Ground station antenna and the servo motors to rotate the structure to the final angles of elevation and bearing.*

4.2 Theoretical simulations

Once implemented the theoretical algorithms and methods to perform the steering capabilities in both ends, it has been carried out a simulation to verify that the algorithm is working as expected.

For this experiment, it has been simulated the coordinates of a typical route for an UAV, shown in Fig. 4.3, in order to compute the angles of elevation and bearing of both antennas and check their rightness.

For this case, the ground station antenna is suposed to be placed in a specific ground location with fix coordinates of (61° 27' 10.3" N, 23° 51' 59" E) and an altitude of 2 [m].

As commented before, the algorithm is installed in MAVProxy and MAVLink, which is installed at the same time in the BeagleBone integrated in the UAV. This BeagleBone can be accessed with the secure shell (SSH) protocol to start the simulation. In this specific application, we are using the program 'Putty' to establish this cryptographic network protocol.

After running the algorithm made during this thesis, we obtained the results shown in Table 4.1.

It is important to remember the parameters playing a role in the computation of the angles:

Figure 4.3 *Fragment of map showing a set of coordinates used to simulate a typical route for an UAV. The figure is extracted from Google Maps with its permission.*

i $\phi_1 \equiv Latitude\ of\ the\ UAV\ in\ radians$

ii $\phi_2 \equiv Latitude\ of\ the\ ground\ station\ antenna\ in\ radians$

iii $\lambda_1 \equiv Longitude\ of\ the\ UAV\ in\ radians$

iv $\lambda_1 \equiv Longitude\ of\ the\ ground\ station\ antenna\ in\ radians$

After studying the results and checking them manually, it is concluded that the theoretical design of the steering system is giving the final results and values properly. The algorithm is sending the angles shown in the table to rotate the servo motors to the desired angles.

4.3 Empirical measurements

Once checked the theoretical algorithm designed during this thesis works, real tests in real scenarios have been carried out to see how the system works in real situations and to make an estimation the error of the algorithm.

Table 4.1 *Results obtained for the angles of elevation and azimuth of both antennas after running the algorithm with simulated waypoints (WP).*

Waypoint ID	**Coordinates of the UAV**	**Altitude UAV [m]**	ξ_1	θ_1	ξ_2	θ_2
1	61° 27' 10.0" N 23° 51' 58.9" E	5	0.1489°	189.0508°	89.8510°	9.0500°
2	61° 27' 9.6" N 23° 51' 58.7" E	5	0.1351°	191.5744°	89.8649°	11.5744°
3	61° 27' 9.2" N 23° 51' 58.5" E	10	0.2261°	192.2553°	89.7738°	12.25528°
4	61° 27' 9.1" N 23° 51' 58.1" E	10	0.2004°	199.7183°	89.7995°	19.7183°
5	61° 27' 8.9" N 23° 51' 57.6" E	10	0.1653°	205.5422°	89.8346°	25.5422°
6	61° 27' 8.7" N 23° 51' 56.8" E	15	0.2164°	213.308°	89.7835°	33.3083°
7	61° 27' 8.7" N 23° 51' 56.2" E	15	0.1991°	219.9052°	89.8008°	39.9052°
8	61° 27' 8.7" N 23° 51' 55.5" E	15	0.1798°	226.2702°	89.8201°	46.2702°
9	61° 27' 8.9" N 23° 51' 55.1" E	15	0.1786°	233.0867°	89.8213°	53.0867°
10	61° 27' 9.2" N 23° 51' 54.7" E	15	0.1785°	241.8391°	89.8213°	61.8391°
11	61° 27' 9.4" N 23° 51' 54.5" E	15	0.1786°	247.2894°	89.8212°	67.2894°
12	61° 27' 9.8" N 23° 51' 54.5" E	15	0.1167°	256.9102°	89.8832°	76.9103°
13	61° 27' 10.0" N 23° 51' 54.7" E	15	0.1241°	261.6933°	89.8759°	81.6933°
14	61° 27' 10.2" N 23° 51' 55.3" E	15	0.1452°	266.7625°	89.8547°	86.7625°
15	61° 27' 10.4" N 23° 51' 55.8" E	10	0.1674°	273.7397°	89.8325°	93.7394°
16	61° 27' 10.5" N 23° 51' 56.3" E	10	0.1958°	278.8108°	89.8041°	98.8108°
17	61° 27' 10.7" N 23° 51' 57.1" E	10	0.2553°	293.7752°	89.7446°	113.7752°
18	61° 27' 10.6" N 23° 51' 57.8" E	5	0.1499°	297.6158°	89.8510°	117.6158°
19	61° 27' 10.5" N 23° 51' 58.5" E	5	0.3022°	309.9303°	89.6978°	129.9302°

The measurements and tests have been performing by fixing the position of the ground station antenna, while the on-board antenna is moving to a set of different coordinates. By doing so, the pointing angles of the algorithm can be characterized, along with the levels of power received and levels of SNR in each situation. All the test have been carried out with LOS[1].

To characterize the error of the algorithm, the on-board antenna has been programmed to sweep a region of space of 180°. Se servo motors pre-configured before allow us to rotate the antenna from 0° to 180° easily.

At the same time, another script is going to be executed in the on-board antenna, this script is the one that manages the reading of the different values regarding receiver power, instantaneous losses, level of SNR, etc. With this, we can obtain these parameters for every degree swept.

Once obtained this reference curve, we can start the comparison between the same values, but obtained with the pointing algorithm, and finally an estimation of this error. To do so, the antenna has been pointed with the algorithm and twenty measurements have been taken in this point.

In Fig. 4.4 is presented the scenario in which the measurements have been carried out. It is important to see the buildings and obstacles around to explain some unexpected situations in the measurements done, explained below.

Once presented the setup chosen for the measurements and the map showing different buildings and obstacles, let's present the final results obtained and make conclusions about them.

The measured power received shown in the plot is presented as a normalized value, to make easier the comprehension of the results.

4.3.1 Test 1

The values obtained with this test are shown in Fig. 4.5. The measurements have been done in approximately 300 [m] with LOS.

The blue curve in all the tests represents a Direction of Arrival (DOA) measurement made with the servos, from 0° to 180°. The inspiration of this plan has been obtained from the method and measurements done in [28].

[1]Line of Sight

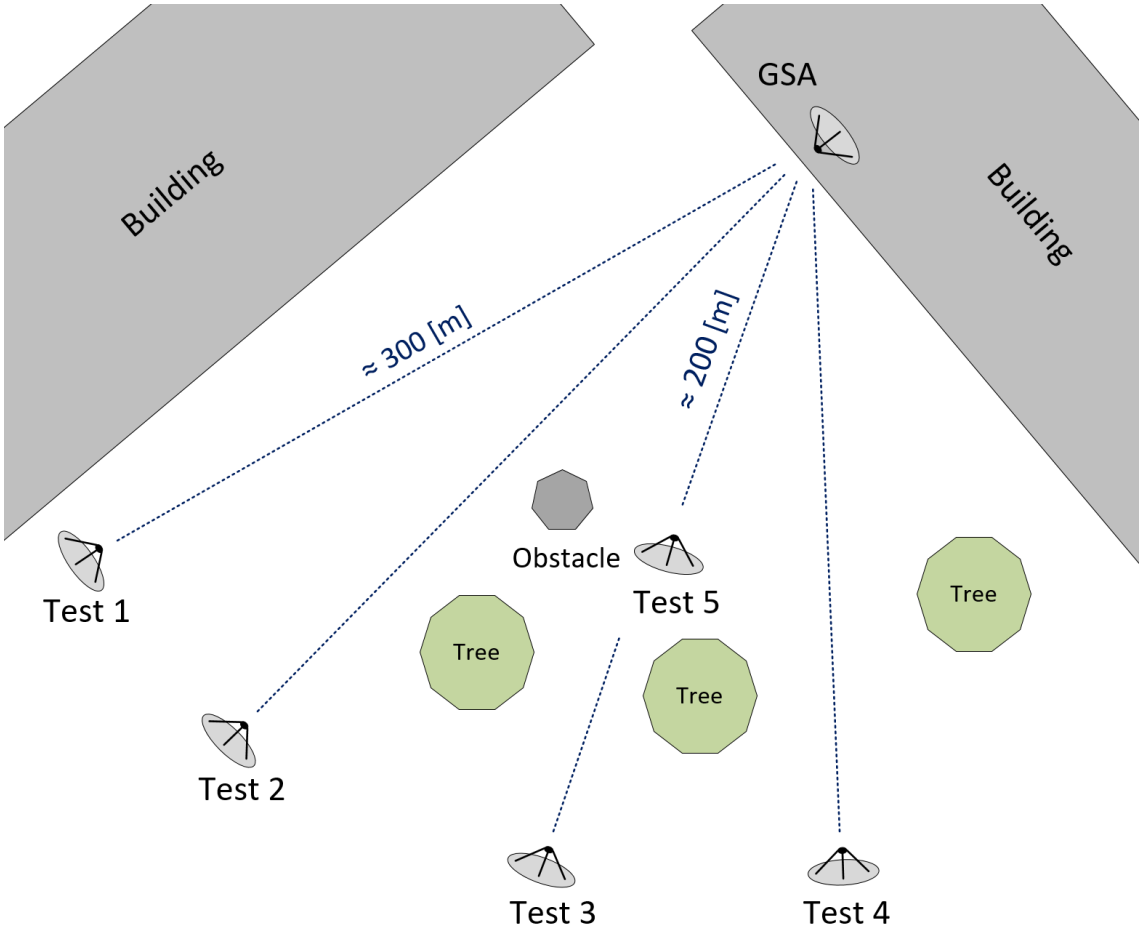


Figure 4.4 *Fragment of map showing the different locations used to perform the real measurements.*

The main normalized values are shown in Table 4.2.

Table 4.2 *Numerical values of the Test 1.*

Parameter	**Normalized Value**
Maximum sweep power received	0 [dBm]
Maximum algorithm power received	-1 [dBm]
Minimum algorithm power received	-13 [dBm]
Mean algorithm power received (μ)	-3.98
Algorithm standard deviation (σ)	3.34

We can conclude that with the steering algorithm we are receiving an average level of power of 3.98 [dB] below the maximum power received with the sweep measurement.

Note this is a restrictive criterion, as we as considering the very maximum of the receiving power.

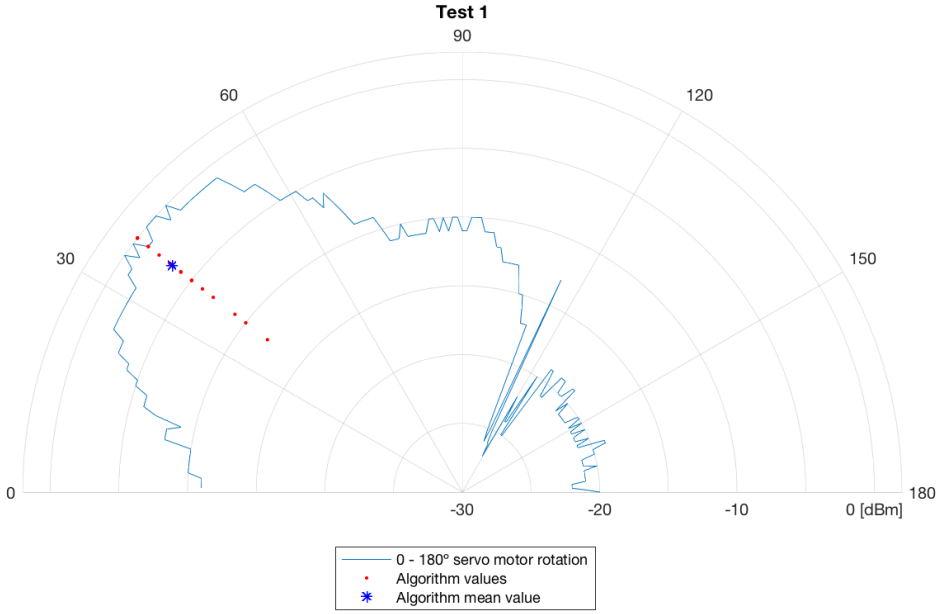


Figure 4.5 *Test 1: Levels of power measured by performing a sweep from 0° to 180° and levels of power measured with the algorithm.*

Additionally, the probability density function of the values received with the algorithm has been computed. This function allows to understand how far the values of the received power with the algorithm tend to be with respect the mean value.

The characterization can be computed by using a normal function with the corresponding values of μ and σ, with the form in Equation 4.1:

$$f(x) = \frac{1}{\sigma \cdot \sqrt{2 \cdot pi}} \cdot e^{\frac{-(x-\mu)^2}{2 \cdot (\sigma)^2}}, \tag{4.1}$$

where μ represents the mean value of the function and σ refers to the standard deviation.

The probability density function (PDF) for this first test is shown in Fig. 4.6.

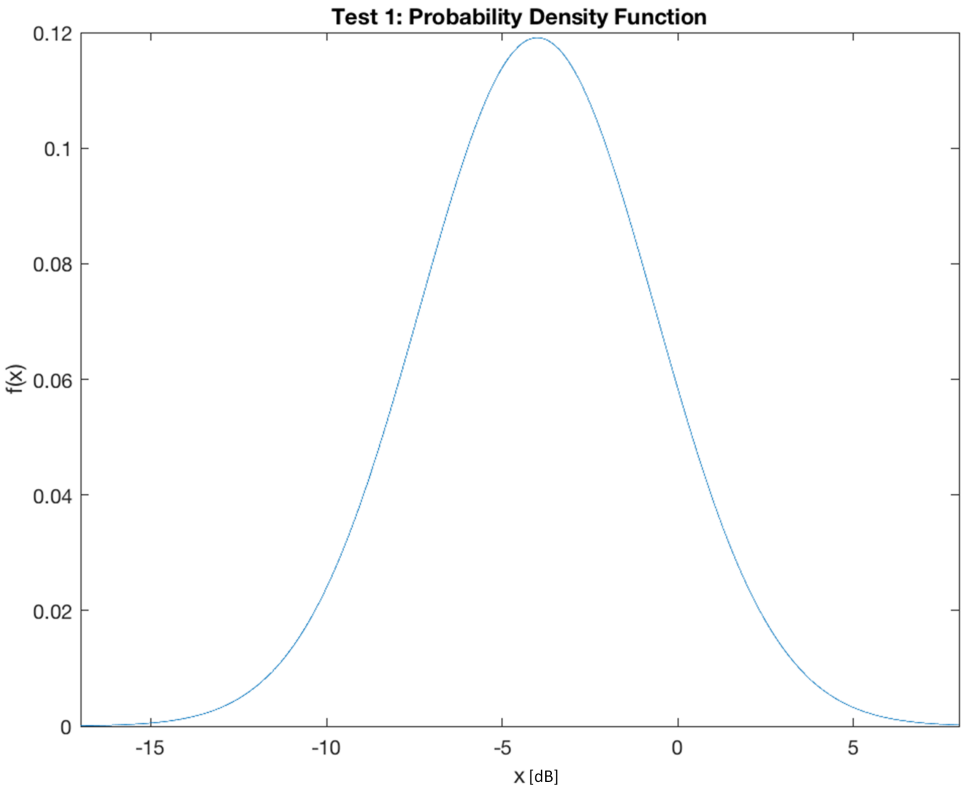


Figure 4.6 *Test 1: Probability Density Function of the twenty values of measured received power by using the steering algorithm. The function $f(x)$ represents the probability of the measurements of having a certain value x.*

4.3.2 Test 2

The results obtained with this test are shown in Fig. 4.7. The test is done at a distance of approximately 300 [m] and LOS.

We can clearly see in the plot above the effects of multipath. In the main lobe of the antenna, where it is supposed to be its maximum value, two values appear with received power levels of -17 [dB] and -11 [dB] with respect its maximum value. This could mean that different reflexions of the signal and multipath are added negatively in the receiving antenna, causing an instantaneous fading in the received signal, as shown in [29].

The main normalized values are shown in Table 4.3.

In this second case, the averaged received power with the steering algorithm is 4 [dB] below the maximum power received with the sweep measurement.

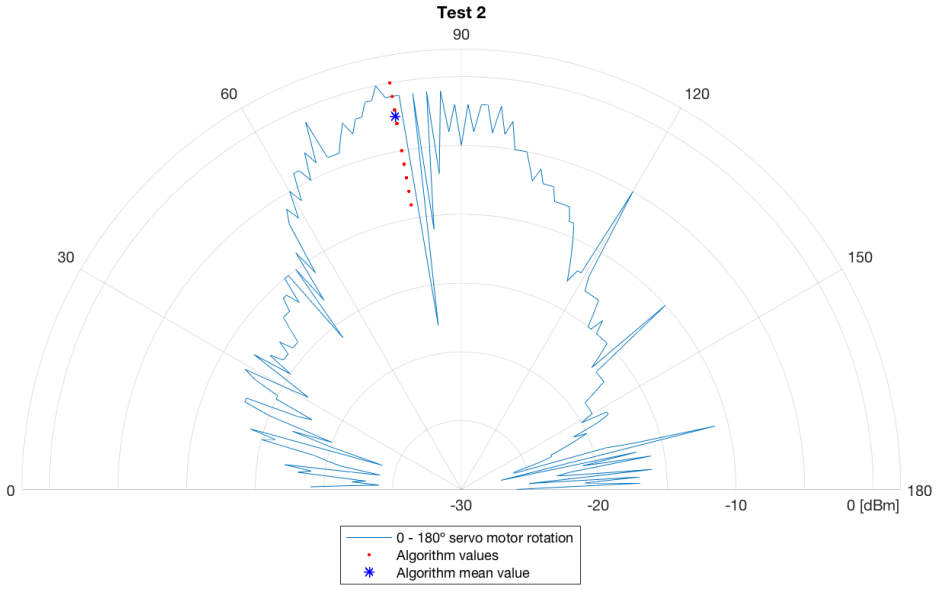


Figure 4.7 *Test 2: Levels of power measured with a sweep from 0° to 180° and levels of power measured with the algorithm.*

Table 4.3 *Numerical values of the Test 2.*

Parameter	**Normalized Value**
Maximum sweep power received	0 [dBm]
Maximum algorithm power received	0 [dBm]
Minimum algorithm power received	-9 [dBm]
Mean algorithm power received (μ)	-4
Algorithm standard deviation (σ)	2.69

The PDF of the values obtained with the algorithm is shown in Fig. 4.8.

4.3.3 Test 3

The empirical results of this test are shown in Fig. 4.9. The measurements have been carried with LOS.

As we can see in the plot above, the steering algorithm pointed the antenna with a misalignment of 1 or 2°, not hitting the maximum of the received power, this could happen due to some inaccuracy in the pointing structure, not been able to rotate to the angle given by the algorithm. Despite this slight misalignment, the values of the power received are close to the maximum value obtained by the reference

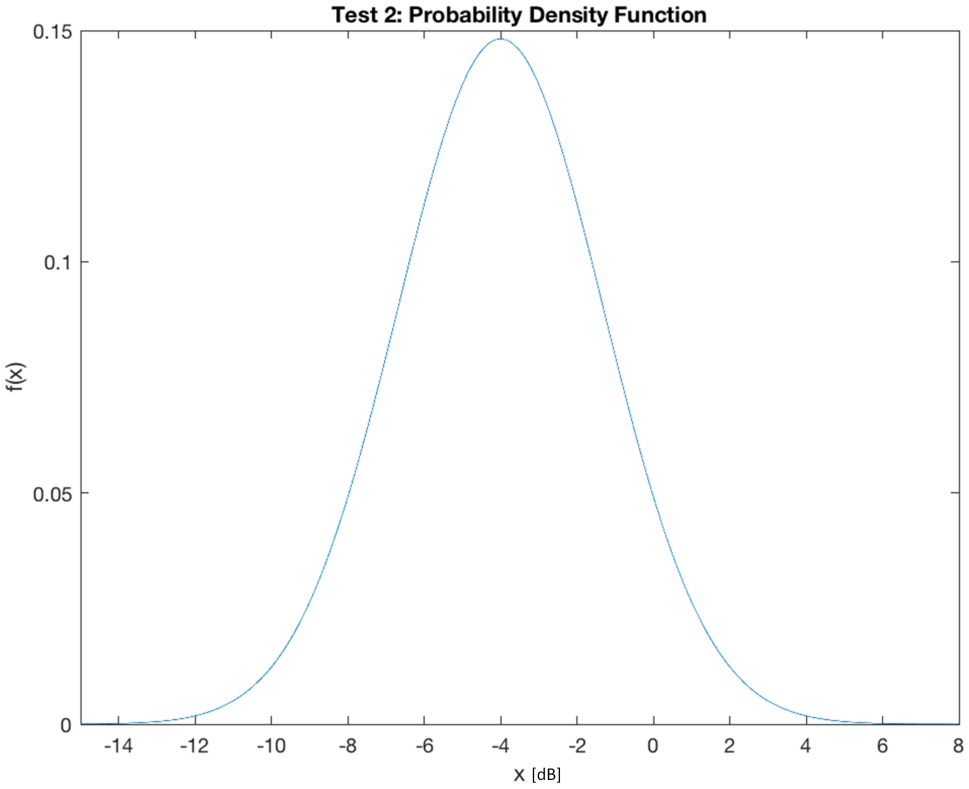


Figure 4.8 *Test 2: Probability Density Function of the twenty values of measured received power by using the steering algorithm. The function $f(x)$ represents the probability of the measurements of having a certain value x.*

measurements.

The main normalized values are shown in Table 4.4.

Table 4.4 *Numerical values of the Test 3.*

Parameter	**Normalized Value**
Maximum sweep power received	0 [dBm]
Maximum algorithm power received	0 [dBm]
Minimum algorithm power received	-10 [dBm]
Mean algorithm power received (μ)	-2
Algorithm standard deviation (σ)	2.11

In this test, the received power drops 2 [dB] with respect the maximum received power with the sweep measurement.

The PDF of the values of received power with the algorithm is shown in Fig. 4.10.

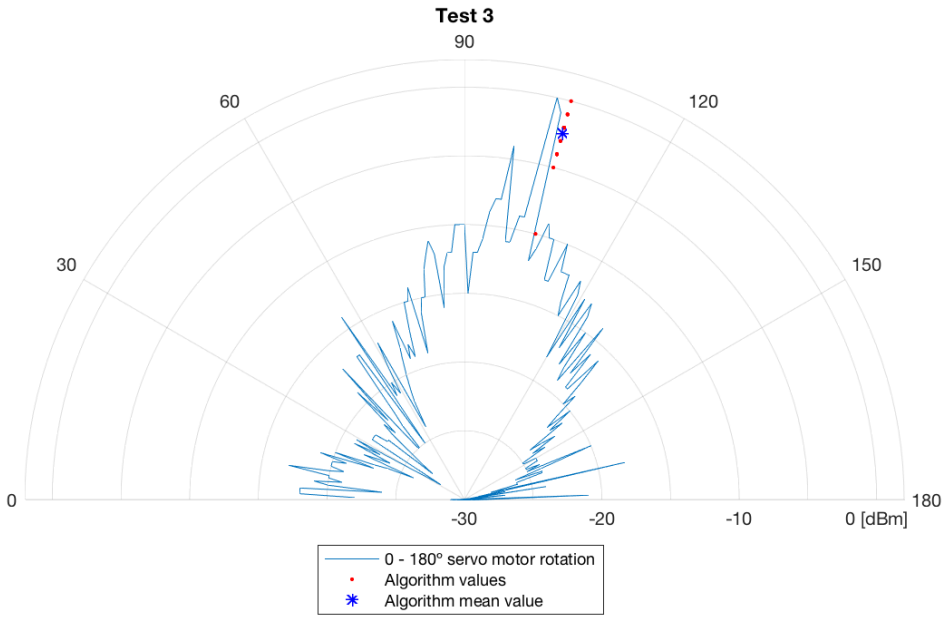


Figure 4.9 *Test 3: Levels of power measured with a sweep from 0° to 180° and levels of power measured with the algorithm.*

4.3.4 Test4

The results of the test 4 are shown in Fig. 4.11. The measurements have been carried with direct LOS.

The main normalized values are shown in Table 4.5.

Table 4.5 *Numerical values of the Test 4.*

Parameter	**Normalized Value**
Maximum sweep power received	0 [dBm]
Maximum algorithm power received	0 [dBm]
Minimum algorithm power received	-6 [dBm]
Mean algorithm power received (μ)	-3
Algorithm standard deviation (σ)	2.2

In this test, the average received power with the algorithm is 3 [dB] below the maximum reference received power.

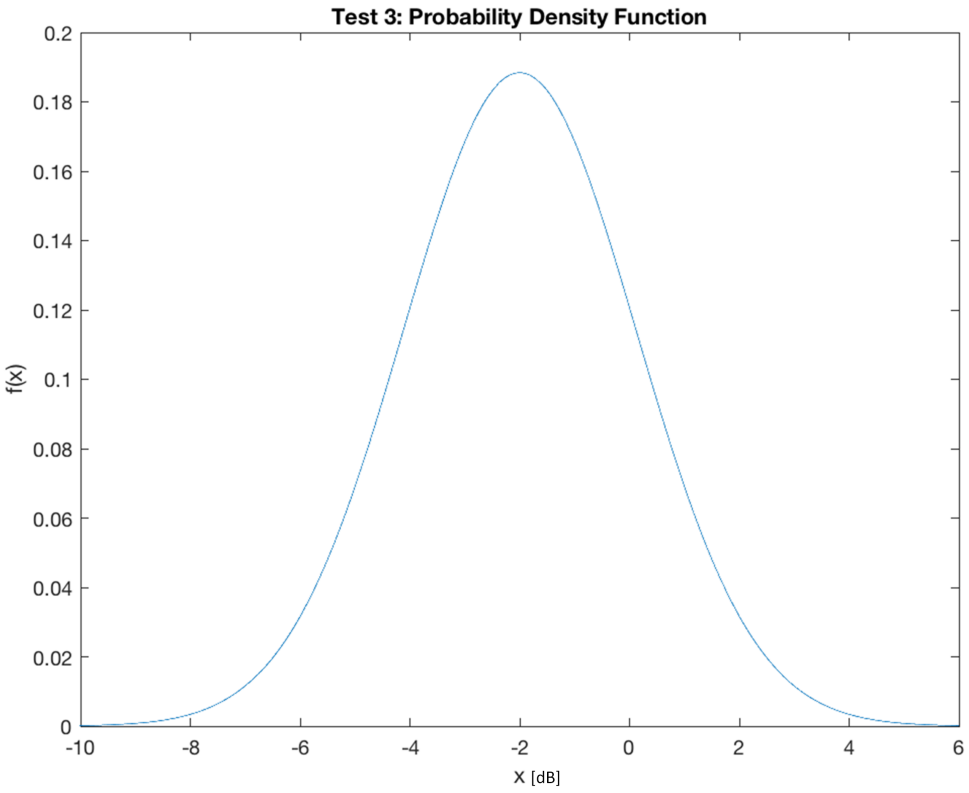


Figure 4.10 *Test 3: Probability Density Function of the twenty values of received power measured by using the steering algorithm. The function $f(x)$ represents the probability of the measurements of having a certain value x.*

The probability density function of the values is shown in Fig. 4.12.

4.3.5 Test 5

The empirical results of test 5 are shown in Fig. 4.13. The distance of the antennas in this measurements is around 200 [m], with LOS.

It can be seen from the results the effect of multipath in the left side of the plot, where the curve of power received changes abruptly compared to the right part, where it changes smoothly. This could be triggered by the obstacle situated in the left side of the antenna when performing the measurements.

The normalized values regarding this test are shown in Table 4.6.

Therefore, the average received power when executing the algorithm is 3 [dB] below the reference power measured with the sweep.

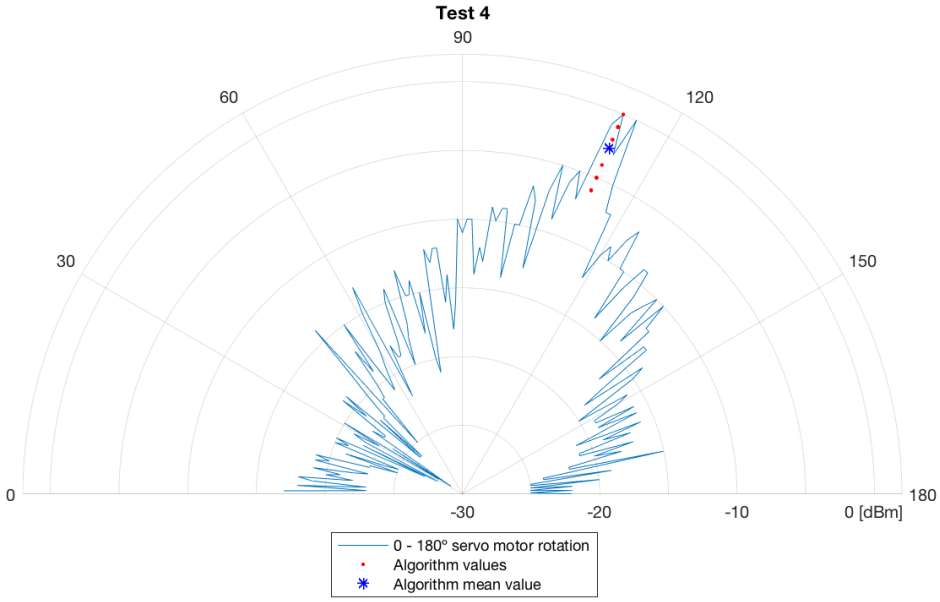


Figure 4.11 *Test 4: Levels of power measured with a sweep from 0° to 180° and levels of power measured with the algorithm.*

Table 4.6 *Numerical values of the Test 5.*

Parameter	**Normalized Value**
Maximum sweep power received	0 [dBm]
Maximum algorithm power received	0 [dBm]
Minimum algorithm power received	-11 [dBm]
Mean algorithm power received (μ)	-3
Algorithm standard deviation (σ)	-2.8

The probability density function is presented in Fig. 4.14.

4.3.6 Comparison of the results

In this final section, all the results obtained in the tests 1 to 5 are presented and compared between them. By doing this, we will get a better understanding of the behaviour of the algorithm.

Firstly, results related with receiver power have been studied and compared. The

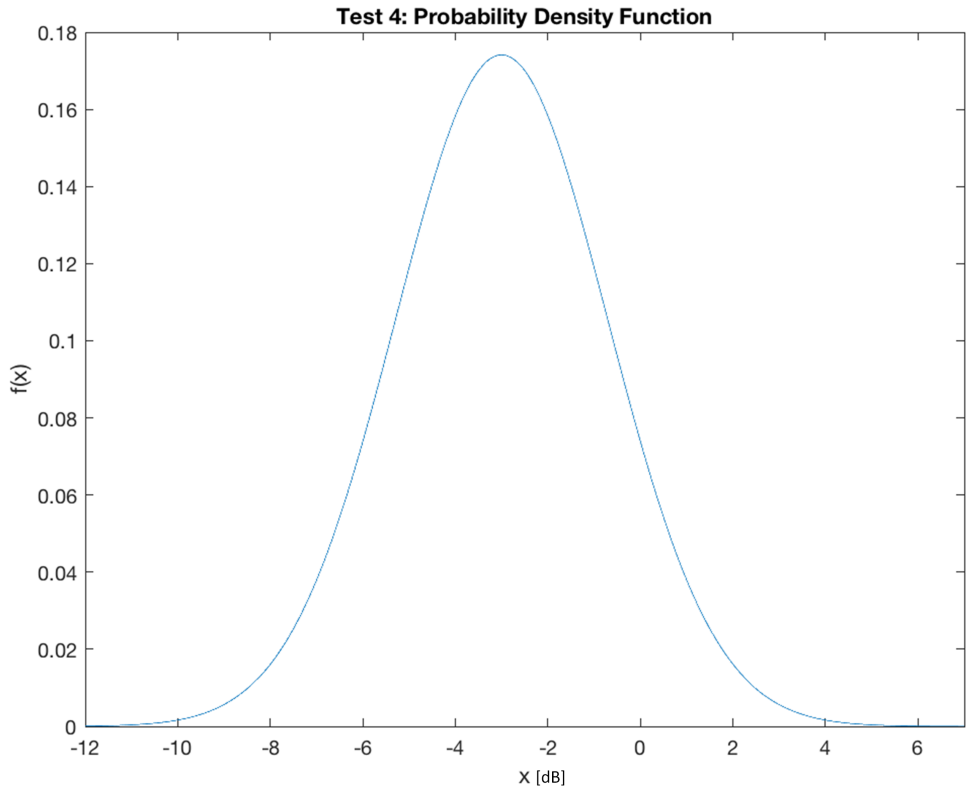


Figure 4.12 *Test 4: Probability Density Function of the twenty values of received power measured by using the steering algorithm. The function $f(x)$ represents the probability of the measurements of having a certain value x.*

comparison includes the next values:

- Ideal value of the received power.
- Maximum measured value of the sweep measurement.
- Mean value measured when using the algorithm.

Let's first compute the ideal power received that we should have in the receiver antenna if a perfect scenario is considered. The received power can be obtained from the Friis formula, that reads:

$$P_{Rx,ideal}\,[dBm] = P_{Tx} + G_{Tx} - L_{BF} + G_{Rx}, \tag{4.2}$$

where the transmitted power is $P_{Tx} = 20\,[dBm]$, the gain of the transmitter antenna

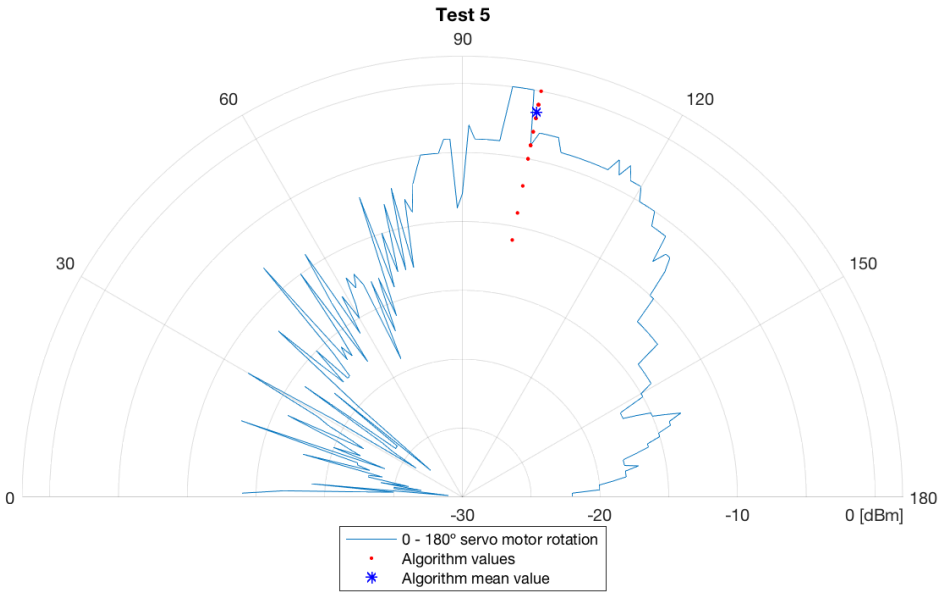


Figure 4.13 *Test 5: Levels of power measured with a sweep from 0° to 180° and levels of power measured with the algorithm.*

is $G_{Tx} = 24.5\,[dBi]$, the gain of the receiver antenna is $G_{Rx} = 16\,[dBi]$, and the free space losses are computed as:

$$L_{BF}\,[dB] = 32.45 + 20 \cdot \log(f\,[MHz]) + 20 \cdot \log(d\,[km]), \tag{4.3}$$

where the frequency is $f = 5880\,[MHz]$ and the distance is $d = 0.3\,[km]$.

Secondly, results related with the signal to noise ratio (SNR) are presented. The comparison includes the next parameters:

- Ideal value of the SNR.
- Maximum measured value of the sweep measurement.
- Mean value measured using the algorithm.

For the calculations of the ideal SNR, we need the ideal receiver power computed previously. By having this parameter, the signal to noise ratio can be computed directly as in Equation 4.4:

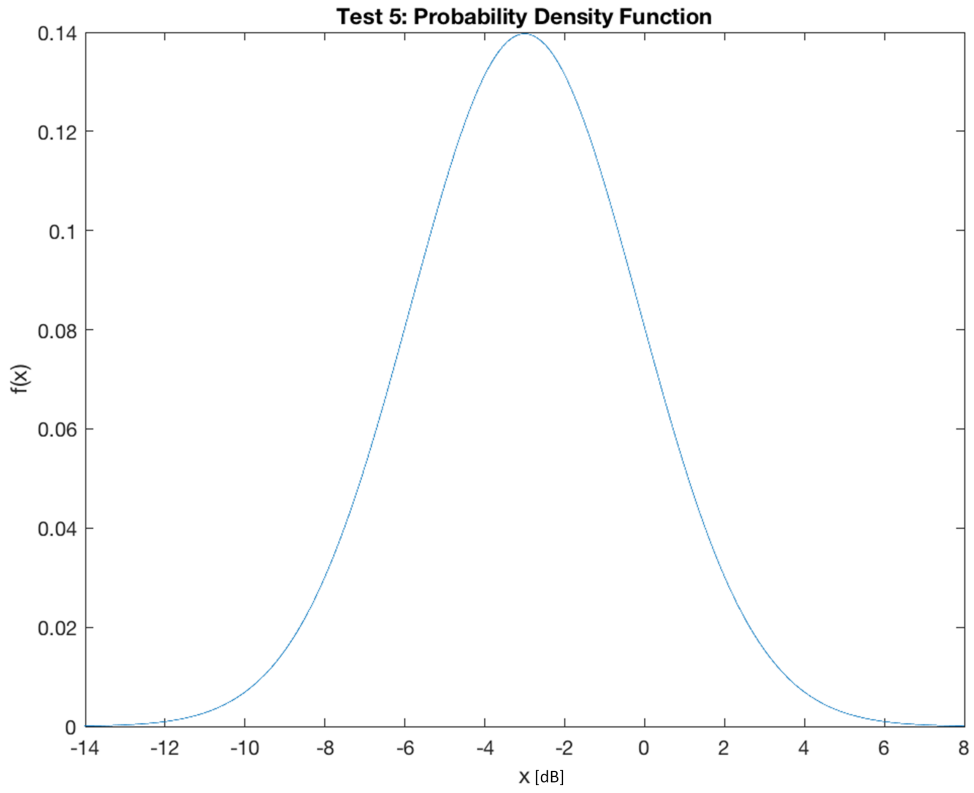


Figure 4.14 *Test 5: Probability Density Function of the twenty values of received power measured by using the steering algorithm. The function $f(x)$ represents the probability of the measurements of having a certain value x.*

$$SNR_{Ideal}\,[dB] = P_{Rx,ideal} - N, \tag{4.4}$$

where the noise power in the receiver is calculated as:

$$N\,[dBm] = k_B \cdot T \cdot B \cdot F = 10 \cdot \log(k_B \cdot T \cdot B) + 1 + 30, \tag{4.5}$$

where $k_B = 1.38064852 \cdot 10 - 23\,[m^2\,kg\,s^{-2}\,K^{-1}]$ is the Boltzman constant, $T = 258\,[K]$ corresponds to the temperature, $B = 40 \cdot 10^6\,[MHz]$ refers to the bandwidth, and $F = 1\,[dB]$ is the noise figure of the receptor.

Note that 30 [dB] are added to the final result to convert from [dBW] to [dBm].

Finally, a table comparing all the result is presented in Table 4.7.

It is interesting to comment that the received power in the fifth test is even bigger

Table 4.7 *Comparison of the values obtained in tests 1 - 5.*

Measurement Concept		**Received power [dBm]**	**SNR [dB]**
Ideal calculated values		-35.29	73.7
Test 1	Sweep measurement (max. value)	-41	68
	Algorithm (mean value)	-45.19	63.19
Test 2	Sweep measurement (max. value)	-37	72
	Algorithm (mean value)	-43	66
Test 3	Sweep measurement (max. value)	-35	74
	Algorithm (mean value)	-39.43	69.57
Test 4	Sweep measurement (max. value)	-39	70
	Algorithm (mean value)	-39.7	69.3
Test 5	Sweep measurement (max. value)	-32	77
	Algorithm (mean value)	-33.55	75.45

than the ideal value. This can be caused due to multipath effect, affecting the signal positively and increasing its level in the received antenna.

With these tests, the error of the algorithm can be characterized and estimated, allowing us to have a better understanding of how our algorithm is performing.

In the next chapter, potential future lines to keep developing and studying the project are presented. These lines explain possible improvements to be done and implemented with the system to enrich the overall system. Also some new functionality to be added is also explained, offering new possibilities and advancements.

5. FUTURE WORK

At this point of the document, a first version for the antenna steering algorithm with servo motor rotation has been designed. This system is capable of rotate the servo motors in the specific angles calculated by the steering algorithm. However, as this version is still a prototype, and there are a lot of aspects that could be improved or new functionality to be added that can enrich the system substantially.

In this section, several ideas regarding future work or future lines of the system are presented. These next lines present the next steps to keep developing and improving the system for its final professional use.

First of all, in Appendix B, it is presented the hardware setup done during this project for the aerial vehicle and the ground station. Although this setup has not been used at the end for obtaining the final results, it can be taken as a basis to begin building the prototypes of this devices.

5.1 Backbone link

First of all, let's depict future ideas and possibilities of how the system can obtain the most of the backbone link.

All the implementations and tests done until now has been designed to work at small-medium distances. Notwithstanding, the final purpose of the project is to use a backbone link working in medium-high distances. To make this possible, several approaches has been considered:

- Improve the hardware equipment in the UAV's end and ground station: The easiest thing to increase the maximum distance of the backbone link is to have more powerful equipment installed in both UAV and ground station antenna. By buying antennas with more gain and more directivity, we could reach much larger distances for this link.

  Remember that with more directive antennas, the error margin at the time of

pointing them is reduced, so we will also need a more precise steer structure that needs to point the devices accurately to each other.

- Study and characterize several propagation channels of the flying vehicle for different conditions, as proposed by the authors in [30] and [31].

- Design a system to improve the flight-time of the UAV and improve its effective use and performance, as explained in [32]. In [33], another algorithm is proposed to increase this battery life-time of the flying vehicle. The final purpose of the system is to provide temporary coverage to certain areas. To do it successfully, we need the UAVs to have enough battery life-time to maintain it flying above the area. Several approaches have been considered to reach this goal:

  - Switch between two UAV with several batteries: One UAV flying while the other waits landed. When the first UAV starts running out of battery, the second UAV is launched and the first one is launched and its batteries are replaced. This way, the communication is never lost when switching between UAVs.
  - Several batteries mounted in the UAV and switched autonomously. Another solution could be to place several batteries in the UAV and switch automatically between them by using an electrical system, for example a switch.
  - Use fuel engines for the UAV: Fuel engines for UAVs started to develop a few years ago. This solution can be used in our UAV to provide 1 - 3 [h] of flight time. Typically, this engines are capable to provide 3 [hp] by using two-cylinder motors combined with spark plug and direct injection of a kerosene solution as the fuel [34].

    With this solution, the UAV can be refueled easily and rapidly to continue giving the communication between base station and users.

- Configure a fleet of UAVs communicating all together to enhance the potential use of the network: Instead of having just one UAV flying at the same time, we could have a fleet of them communicating in real time. By having this configuration, we can course higher throughputs through the network, improving the rates for the final users.

  There are many ways to configure this fleet of flying vehicles, cheaper or more expensive depending on the ambition of the solution.

  - One proposal is to have different ground station antennas communicating with the UAVs, each one implementing its own backbone link. The

advantage of this solution is that the throughput is doubled, as we have more than one backbone link. The problem with this solution is the high associated prize of establishing one extra backbone link.

  - Another interesting proposal is to use just one ground station antenna establishing just one backbone link with a 'master UAV', carrying sectorized antenna to provide coverage in 360°. Around the master drone, we can set a constellation of slave UAVs connecting to it. This way, we only need one backbone link (which makes the final system cheaper) and the steering system implemented in this thesis in each slave UAV to properly point to the master UAV.
  - Another simpler proposal is to use UAVs as relays, carrying the information through the flying vehicles until it reaches the ground station. This solution can be seen as a smart grid of UAVs communicating and interchanging information between them.

- Multiple input multiple output (MIMO) technology has become an essential mechanism to enhance the coverage of wireless networks. A future line for the project could be how this recent technology could affect air to ground communications, as proposed by the authors in [35].

5.2 Steering system

Let's depict now some improvements that could be performed within the proper steering system implemented in the UAV and/or ground station antenna. This changes will be mainly oriented to maximize the accuracy of the steering system.

- Build a precise mechanical structure to integrate the servo motors with the antennas. This will allow to apply these previously calculated angles to the antennas.
- Perform a more precise pointing algorithm considering a curved-Earth model. With this model, the steering system can get better accuracy when is operated at high distances.
- Improve the hardware equipment of UAV and GS: Replace several hardware modules affecting the tracking system with more precise ones. The modules affecting the accuracy are the GPS and compass in the UAV and GS and barometer sensor in the UAV.

- Make the steering system more precise by obtaining the target UAV's yaw, pitch and roll in real time. Once obtained these values, correct the tilt in the antenna caused by them.

Apart from these possible improvements in the steering system, a reliable failsafe should also be defined and implemented to ensure the communication is reestablished if lost. The procedure is oriented in the same way as in [36], but for UAV instead of UGV (Unmanned Ground Vehicle).

The best solution found to get the link back is:

1. Check if the communication is established every time, save a the last coordinate position of the UAV and transmit it with the rest of variables to the ground station antenna.
2. If the connection is lost, the UAV will fly back to this last position, and from there, its antenna will point to the ground station. At the same time, the antenna in the ground station will point to this last-registered GPS position of the UAV.
3. By doing this failsafe, the antennas will be steered again establishing the communication back. At this point, the mission planned can be continued as it was especified.

These ideas presented in this chapter are some next steps to keep developing the project. They illustrate the importance and the necessity of having a steering system keeping the antennas pointed to each other every time. The tracking system can have other uses apart from this one, so it is interesting to keep studying and developing it for future possibilities.

6. CONCLUSIONS

Unmanned flying technology is a rising matter that is beginning to be used more and more in many different technological fields as time goes by. This new technology provides many advantages and facilities compared to the currently used solutions. It is important to study and research this topic in order to improve the proposals already implemented and discover new potential applications for it.

One new use for unmanned vehicles is the carriage of LTE base stations to provide wireless coverage to a large number of users in certain scenarios. This proposal can replace actual solutions, which are very expensive in comparison or even impossible to deploy in difficult terrains.

The carriage of LTE base stations providing coverage for the users is the final purpose of the overall project studied along this research thesis. To make this solution possible, it is necessary to have a ground station, controlling and managing every aspect within the system, an UAV, and a reliable link to communicate both ends. This link will carry telemetry orders for the UAV and user's data to be delivered/received to/from the core network.

At this point, it is necessary to research this topic to make the backbone link as large and as reliable as possible. To do so, very high-gain and high-directivity antennas in both UAV and ground station are used. This solution will allow to communicate the UAV far away from the ground station antenna, but it also requires the antennas to be pointed to each other all time.

As this research thesis shows, it is necessary to have a steering algorithm managing the tracking of the on-board antenna of the UAV and the ground station's antenna to maintain the communication between both ends all the time. A first step to this steering system is taken in this research thesis, where a steering algorithm with servo motor rotation is studied and designed.

In order to tackle the problem, an initial plan for the network has been configured. After this, the backbone link is established. Once having that, we can communicate both UAV and ground station antenna through this link.

The steering algorithm done for this use case has been designed to work in both ends, communicating with each other in real time. To steer the antennas, the algorithm executes sequentially a set of functions. When the process is ended, it starts again as a loop.

The first set of functions are designed to obtain the final angles to apply to both antennas to point them. They involve several heavy mathematical calculations to manipulate the coordinates of the devices.

The second set of functions are programmed to communicate in real time on-board computer in the UAV and the ground station and interchange the required parameters in both ends. Finally, the third set of functions manage the rotation of servo motors to the angles computed before.

This functions and methods are integrated with the software installed in the on-board computer of the UAV, from which it is executed periodically. The result will be an autonomous steering algorithm with servo rotation providing the desired movement.

In the fourth section regarding the validation of the system, a simulation considering theoretical values and five tests taking into account empiric measurements have been carried out. With the theoretical simulation, we can ensure that our steering algorithm is working as expected, providing the final angles of elevation and bearing in the right way. Through the empirical test, the steering algorithm is tested in the real world, providing the angles expected in a real scenario. Additionally, the levels of received power in the on-board antenna have been characterized, allowing us to compare the ideal values with the ones obtained with our steering software.

At this point, we can say that our steering algorithm with servo rotation is designed and working as expected. There are still many future improvements and enhancements to be applied to this first prototype. Further research and study regarding this topic would be very interesting, as this work can be extrapolated to other projects and technologies, offering many advantages and new interesting use cases for them.

A. POSTER PUBLISHED IN TAKE 5G, HELSINKI, 14/12/2017

In the next page, the poster 'Aerial Base Station with Directional Microwave Backhaul Link' published by the team and I in the workshop 'TAKE 5G', held in Helsinki on 14/12/2017, is presented. In the picture, both scenarios described in the introduction can be seen, showing two different examples and use cases where our project could be used in.

Furthermore, the poster shows the configuration of the prototype chosen for the project, along with the main details regarding technical information of each component.

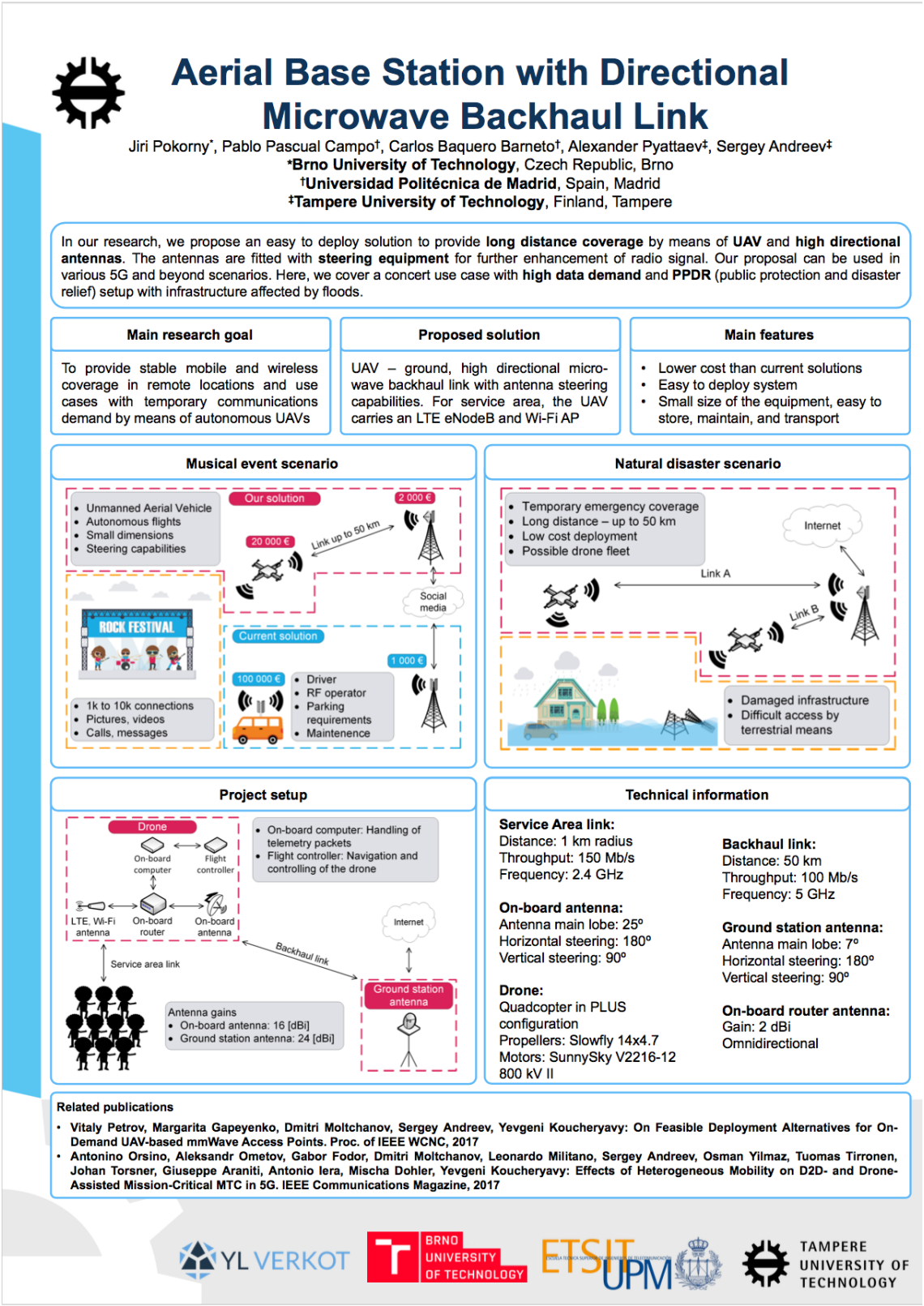


Figure A.1 *Poster published in TAKE 5G workshop, held in Helsinki on 14/12/2017.*

B. HARDWARE SETUP FOR THE UAV AND GROUND STATION ANTENNA

In this appendix, it is presented the hardware setup chosen for the UAV and the ground station. This setup has been implemented along the project, but at the end not used to obtain the final goal and results. However, it can be an excellent point to begin building the physical prototypes of the devices. All the necessary equipment is presented in the following lines, along with its connections and dependencies.

B.1 UAV hardware setup

The main hardware component of our project is the UAV, together with the devices integrated within. The UAV will be flying in a static position providing the backbone link with the ground station and the service link with the users.

The UAV chosen for this application is a quadcopter in PLUS configuration. The original vehicle has been disassembled and built over a lighter frame to gain extra battery lifetime. This custom frame also has several slots to place the required devices more easily.

The hardware prototype of the UAV has been designed similarly to the one made by Byung-Cheol Min, Eric T. Matson and Jin-Woo Jung in their project regarding network enhancement with a robotic system [37]. The difference is that our idea refers to an aerial vehicle instead of a terrestrial one, having another hardware requirements. However, there are still some similarities in both projects.

In the following image, a diagram containing the hardware integrated in the UAV is depicted. Black wires contain information and data transmitted within the drone and red wires represent the feeding of different components.

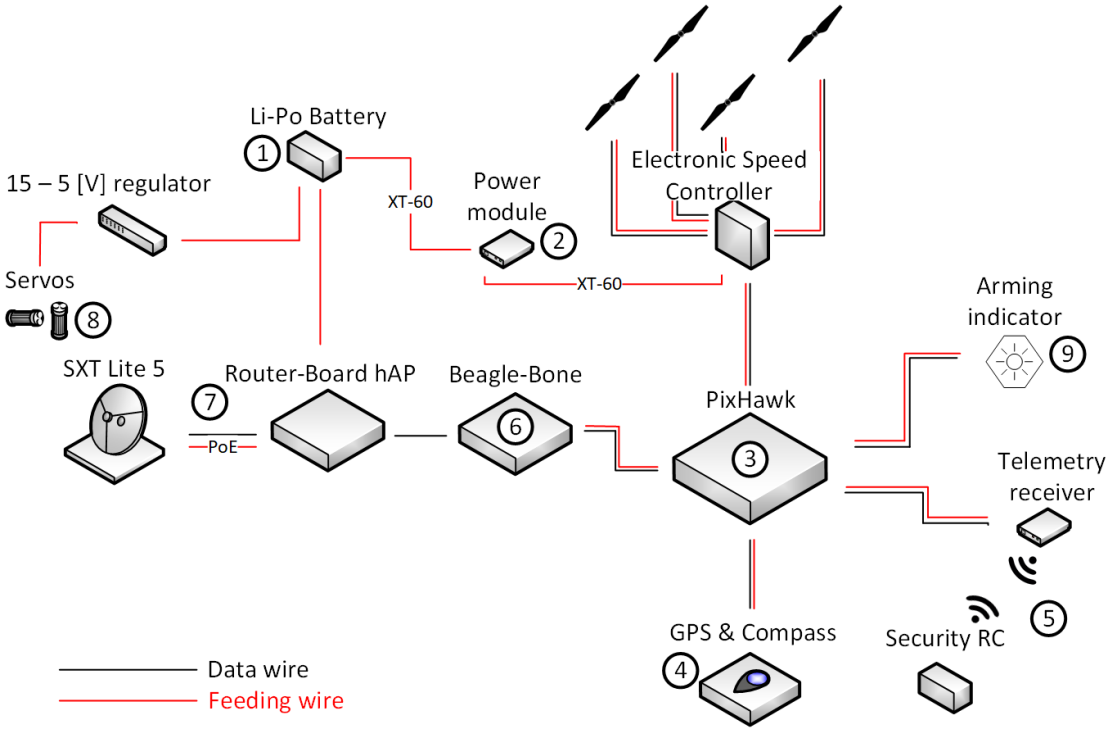


Figure B.1 *UAV hardware setup. Connections between all devices integrated in the UAV are depicted. Black wires refer to data wires and red wires refer to power wires.*

The main hardware components of the UAV are explained in the following lines:

1. Li-Po battery: Feeding device. It provides 15 [V] DC voltage. The battery can storage up to 30,000 [mAh].

2. Power module: Voltage regulator for the UAV. It converts 15 [V] coming from the battery to 5 [V] going to the PixHawk

3. PixHawk: Flight controller. Autopilot integrated module that manages the acceleration of each motor to control the trajectory of the UAV according to its position, its axis state (roll, yaw, pitch) and the orders received by the radio controller (RC) and/or MAVProxy. It also handles the feeding of other devices.

4. GPS & Compass: The module provides the coordinates and heading of the flying vehicle in real time.

5. Telemetry receiver and security RC: Failsafe controller. Security RC acting as a backup in case the backbone link breaks.

   MAVProxy also implements different failsafe measures, in case the communication between the UAV and the ground station breaks. If the antenna

integrated in the drone losses the connection with the antenna in the ground station, the BeagleBone automatically sends a custom failsafe order to the PixHawk. The failsafe order (e.g. Stop the UAV in the current position and wait, land the drone , return to home GPS point, etc.), can vary depending on the application. For our particular use case, if the UAV loose the communication, it will automatically stop and land.

As a future improvement for the project, the failsafe proposed by the authors in [38] could be implemented in our system.

6. BeagleBone: Control station module. The BeagleBone acts as a control station for the UAV. It has MAVProxy installed and sends MAVLink packets to the PixHawk to guide the drone. The BeagleBone is connected to the on-board router and is accessed from the ground station, where the pilot is sending the orders to control the UAV as commands.

7. Router & SXT Lite 5: Used to established the backbone link. Its final mission is to communicate the ground station with the UAV at great distances.

8. Servo motors: These devices provide the steering system for the antenna integrated on the UAV. The model used for the servo motors mounted in the drone is the HEXTRONIX HXT-900[1]. The servo motors are feed directly from the 15 to 5 [V] regulator (see Appendix C).

   All the information about the operation and configuration of the servo motors is detailed in Section 3.4.

9. Arming indicator: Security led to warn that the motors of the UAV are armed.

B.1.1 Weight carried by the UAV

A vital parameter when talking about UAVs is the total weight that is been carried, directly related to the flight time that the battery is capable to provide. Weight of the drone and the hardware should be reduced as much as possible to enlarge this parameter.

To do so, some of the hardware components have been disassembled from its original cases and integrated directly in the frame of the UAV.

In Table B.1 it is presented a list with the total weight that our UAV is carrying, considering all the components integrated within.

[1]See "Servo motor HEXTRONIX HXT-900 web specifications", 2016: https://hobbyking.com/es_es/hxt900-micro-servo-1-6kg-0-12sec-9g.html

Table B.1 *Approximate weight carried by the UAV once all hardware devices have been mounted in the aerial vehicle.*

Hardware device	**Weight [g]**
Drone itself (frame + other hardware)	1,290
Li-Po Battery	600
On-board router	260
On-board antenna	155
Wires	100
BeagleBone	80
Servo motors (x2)	18
Voltage regulator	10
Other elements	150
Approximate total weight	**2,663**

B.1.2 Attachment of the servo motors with the antennas

As a basis for future research, examples for the construction of the mechanical structures to properly rotate the servos are presented. Remember that we need two servo motors for each antenna, one to provide the elevation and the other to provide the azimuth or bearing angle.

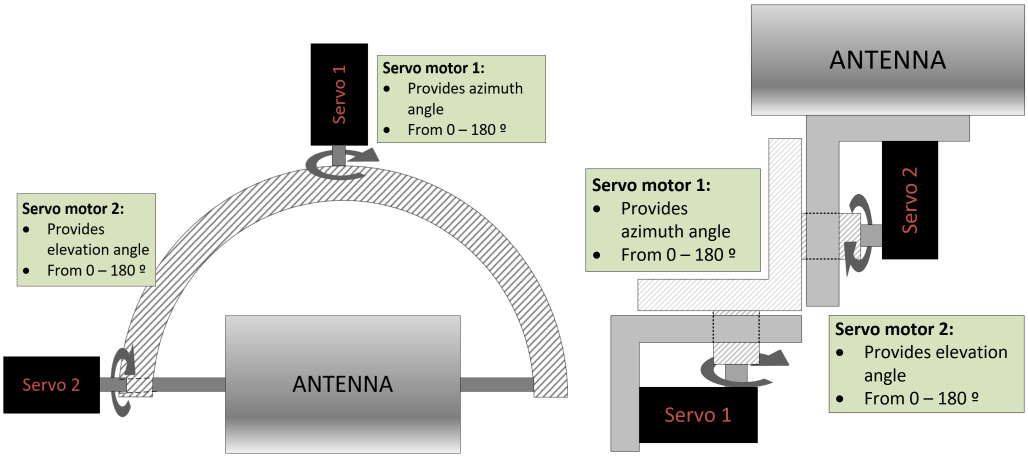
Both structures should have two rotation axis, one for each angle. For the on-board UAV's antenna, a plastic structure could be chosen. Remember that it is very important to reduce as maximum as possible the weight in the UAV to have higher times of battery life. The weight of the structure built is approximately 40 [g].

The structure is going to be attached in the bottom of the UAV, so that no possible obstacle will be placed in front of the antenna distorting its radiation.

The structure built is known as a 'U' shape, capable of providing rotation in both horizontal and vertical axis.

For the ground station antenna, since the weight is not important, a metal structure can be used. The structure should also be capable of provide both elevation and bearing angles. This structure will be typically mounted in a tripod in order to manipulate the height at which the ground station antenna is.

The design for both structures are shown in Fig. B.2, with the setup proposed for the servo motors:



(a) UAV on-board's antenna's shape (b) Ground antenna's shape

Figure B.2 *Shapes built for both antennas to provide rotation in two perpendicular axis, that will correspond to the elevation and bearing angles.*

B.2 Ground station hardware setup

The ground station is mainly composed by a computer and the antenna to establish the backbone link. The computer is connected to the antenna to manage the whole network. It should have installed all the software needed to communicate remotely with the BeagleBone integrated in the UAV (of course, through the backbone link).

The antenna providing the link should preferably be in a high location in order to improve the SNR within the communication. As we are testing the UAV in open field, no external power supply can be accessed, so the equipment in the ground station should be feed with another Li-Po battery.

With this battery and the help of two regulators, all the hardware within the system can be feed. With exception of the antenna, the hardware devices used are not very high power-consuming, so battery-life is not a problem.

In the diagram depicted below, all the connections regarding data and feeding are shown:

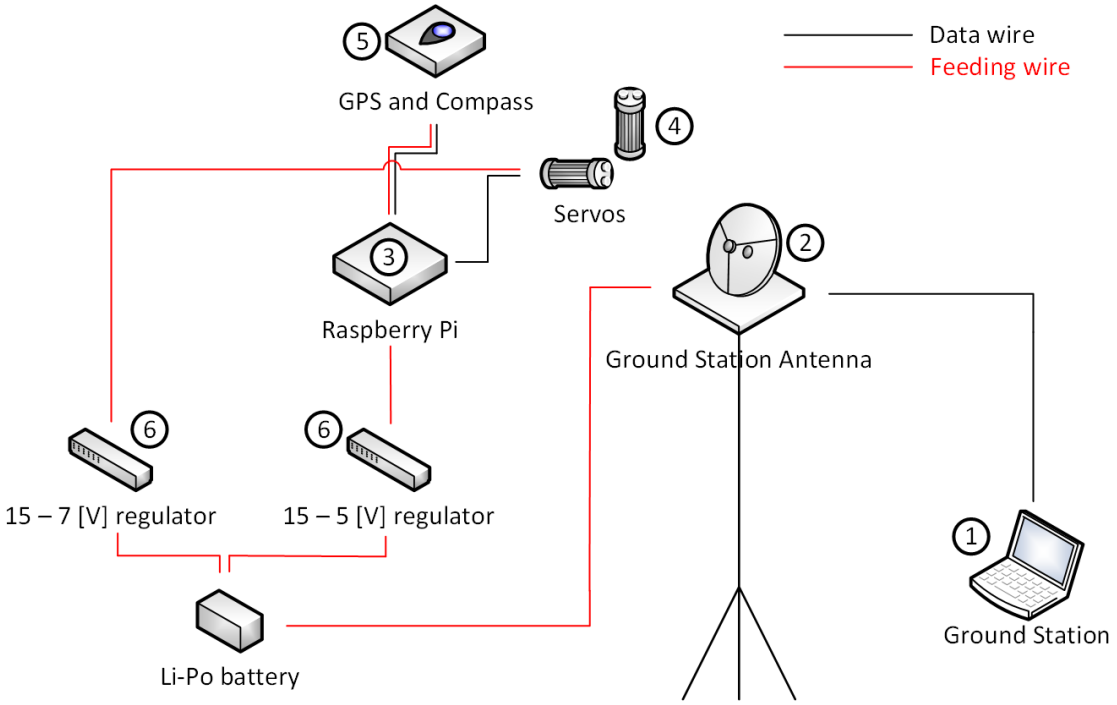


Figure B.3 *GS (Ground Station) hardware setup. Connection between all the devices needed to establish the backbone link. Black wires refer to data wires and red wires refer to power wires.*

1. Ground Station: Computer managing and handling the network in real time. Used to process telemetry packets for the UAV to be sent to the BeagleBone.

2. Ground Station Antenna: Antenna communicating with the UAV's antenna to provide the desired Backbone link.

3. Raspberry Pi: Computer obtaining the GPS and Compass signal and managing the movement of the servo motors. The Raspberry is connected to the ground station to receive orders.

4. Servo motors: Devices providing the steering capabilities in the ground station antenna. As this antenna is way heavier than the one mounted in the UAV, we need other servo motors with higher torque per centimeter. The model used for the servo motors are the CORONA DS339-HV[2].

5. GPS and Compass: Hardware module obtaining the coordinates and heading of the ground station antenna. It should be located as near as possible to the antenna to minimize the error.

[2]See "Servo motor CORONA DS339-HV web specifications", 2017: https://hobbyking.com/en_us/corona-ds339hv-digital-metal-gear-servo-5-1kg-0-13-sec-32g.html

Note the usability of this module if we wanted a moving ground station while performing the steering system (e.g. a parade).

6. Regulators: Explained in Appendix C.

C. ADJUSTABLE VOLTAGE REGULATOR

The devices used during the project require different amounts of voltages and currents applied to their inputs. In the case of the on-board antenna of the UAV, ground station antenna and router, this does not cause any problem because they can be feed directly from the Li-Po batteries. But when we take a look of the required voltages for the servo motors and the Raspberry Pi, we realize that they require a lower voltage in their inputs.

Therefore, these last devices cannot be directly connected to the battery, and need an extra component to down-convert the voltage. This component is called an adjustable voltage regulator.

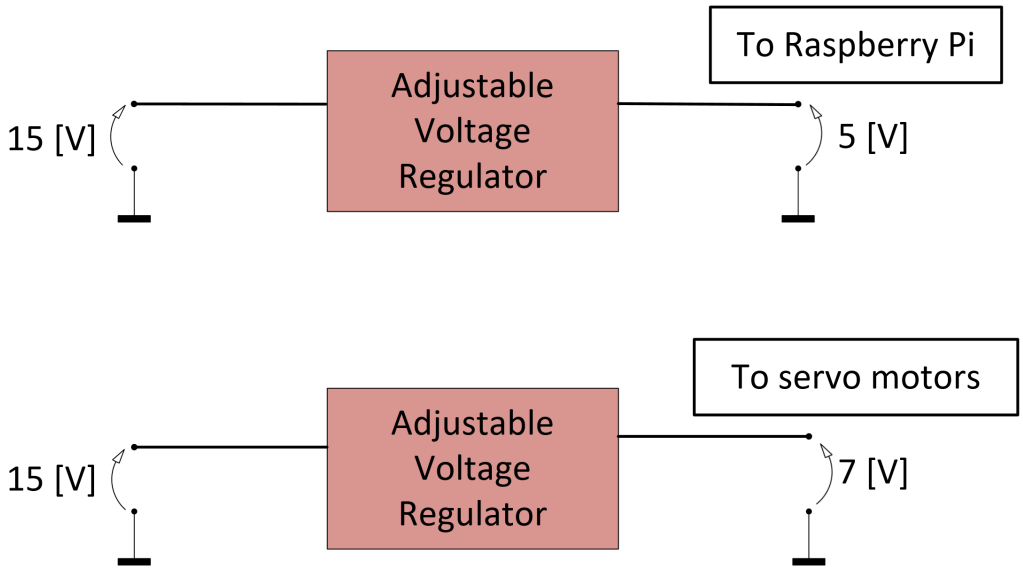


Figure C.1 *General scheme for the regulators.*

To build this component, it has been used the Integrated Circuit (IC) 'LM2575 adj'. Information regarding the component can be found in its datasheet [39]. The IC can provide an adjustable output voltage from 1.2 to 37 [V], with a maximum current of 1 [A].

The component has been built in a universal prototyping printed circuit board

(PCB), considering the specifications and calculations done below.

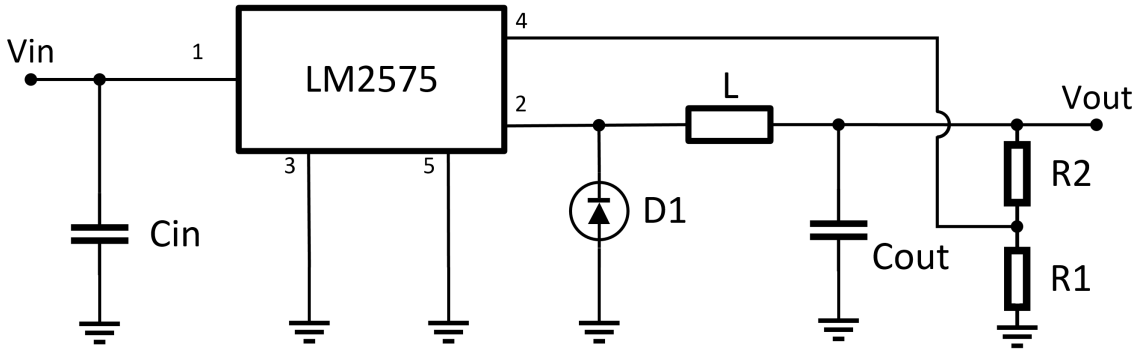


Figure C.2 *Adjustable output voltage regulator model LM2575 equivalent circuit.*

Fig. C.2 shows the squematic of the regulator, where V_{out} represents the regulated output power, $V_{in,max}$ corresponds to the maximum input voltage, $I_{Load,max}$ refers to the maximum current in the load, and $V_{ref} = 1.23\,[V]$ is the reference voltage.

The equation depending on R_1 and R_2 is shown in Equation C.1:

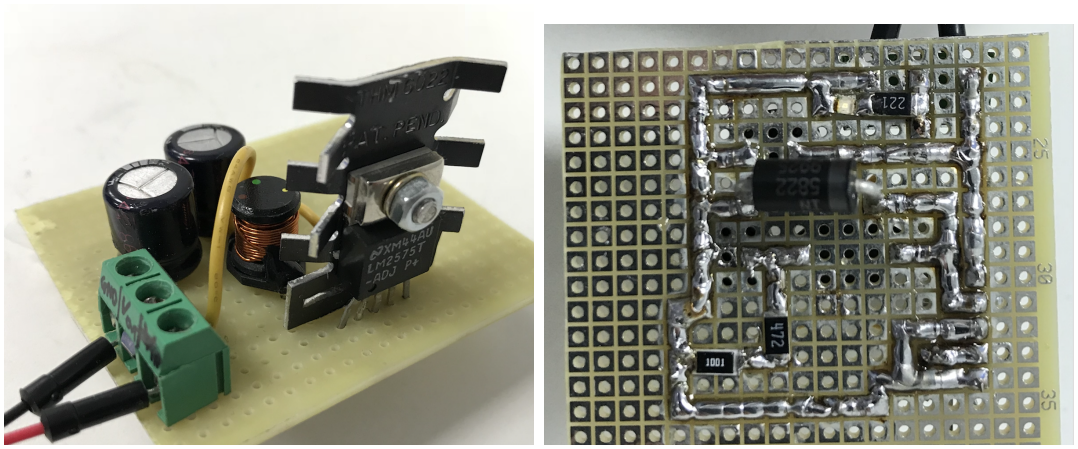
$$V_{out} = V_{ref} \cdot (1 + \frac{R_2}{R_1}). \tag{C.1}$$

Therefore, considering $R_1 = 1\,[k\Omega]$, it is obtained:

For $V_{out} = 5\,[V] \rightarrow\ R_2 = 3\,[k\Omega]$,

For $V_{out} = 7\,[V] \rightarrow\ R_2 = 4.7\,[k\Omega]$.

The final result of the circuit built is shown in Fig. C.3:



(a) Front side (b) Back side

Figure C.3 *Final adjustable output voltage regulator built.*

D. SOCIAL, ENVIRONMENTAL, ECONOMIC, AND PROFESSIONAL ETHIC IMPACT

As engineers, researchers and designers, it is vital to study the impact of the new applications and implementations to be developed in the modern world. It cannot be forgotten that our research is focused on improving people's lives, carrying technology closer to their daily routines to help them in various purposes.

In this appendix, the social, environmental, and economic impact of the the developed work is studied, along with the professional ethics involving the matter.

- Social impact: The final purpose of the developed project is to provide wireless coverage in certain scenarios with variable circumstances. Depending on the scenario to be provided with the service, different advantages can be achieved.

  In the first scenario showed in the introduction regarding a musical event (see Section 1.3.1), the final objective of our project is to interconnect people in a more efficient manner, allowing them to interact between them and communicate easier.

  In the second scenario (see Section 1.3.2), people in distress due to a natural disaster can be helped and assisted faster and in a more reliable way. Integrating the project with the public forces, we can reduce the impact of a natural disaster in the affected area, minimizing its effects and helping the society in this occurs.

- Environmental impact: The environmental impact of our project is clear. By deploying a fleet of UAVs to provide wireless coverage in a certain aera, the telecom operators will avoid the necessity of installing big and bulky base stations in the area under study. All the building process of the base stations, commonly associated with the destruction of the land, deforestation or effect in the landscape can be avoided.

- Economic impact: The economic impact, in this case, can be related with the environmental impact of the project work. The deployment process of a base

station in a certain area is associated with a high cost for the operator and for the government. This problem could be avoided by just deploying the fleet of UAV to provide the temporary coverage in a certain area, with a minimal cost compared to the former solution.

- Professional ethic: According to the professional ethic code for the engineers, the solution proposed should be reliable and safe for the final users. Is for this reason that several failsafes have to be designed for the aerial vehicle, in order to minimize software or mechanical errors in the UAV that could lead to damage of people. Apart from this, the UAV will be flying at a static position in the air far away for the final users, in case any complication is produced.

E. ESTIMATED ECONOMIC BUDGET

For a project to be feasible, it is important to study the estimated prize or cost of the complete develop process. In this section, the estimated costs of the material components and personnel costs are studied.

- Costs associated to the development and design process of the project: The costs regarding the development process can be approximated as follows, considering twelve months as the duration of the development phase of the project.

  1. For the no-material costs, let's consider that five people will be working on the project, one acting as a supervisor and four as developers or workers. It is reasonable to assume a salary of $36,000$ $[\frac{€}{year}]$ for the supervisor and $27,600$ $[\frac{€}{year}]$ for the developers.
     The total sum for personnel costs during the development phase is 542.400 [€].
  2. For the material costs, that means, the cost of the hardware and software equipment, the prize can be estimated in $10,000$ [€], which is the approximate cost of the prototype where the tests have been performed.

- Costs associated to the proper deployment and operation of the system once developed: Once implemented and tested the prototype, the real systems can begin being produced and utilized. This points approximates the cost of the deployment of the system in an specific scenario.

  1. First of all, the cost of the final system, composed mainly by the UAV, ground station, and antenna setup, can be estimated to $20,000$ [€] for each system produced.
  2. Let's assume that there are 4 people working on the system with a salary of $25,000$ $[\frac{€}{year}]$.
  3. Apart from this, the cost of the associated segment to communicate with the core network can be approximated to $2,000$ [€].

F. IMPLEMENTATION OF PYTHON ALGORITHMS

F.1 Algorithm obtaining the elevation and bearing angles of both antennas

```
def antennaTracking(lat1, lon1, alt1, lat2, lon2, alt2):
    # Convertion to radians
    lat1 = lat1 * (math.pi / 180)
    lon1 = lon1 * (math.pi / 180)
    lat2 = lat2 * (math.pi / 180)
    lon2 = lon2 * (math.pi / 180)

    # Bearing angle to rotate the antennas

    # Ground Antenna
    y1 = math.sin(lon2 - lon1) * math.cos(lat2)
    x1 = math.cos(lat1) * math.sin(lat2) - math.sin(lat1
        ) * math.cos(lat2) * math.cos(lon2 - lon1)

    bearing1 = math.atan2(y1, x1)

    bearing1 = bearing1 * (180 / math.pi)

    if (bearing1 < 0):
        bearing1 = bearing1 + 360

    # Drone Antenna
    y2 = math.sin(lon1 - lon2) * math.cos(lat1)
    x2 = math.cos(lat2) * math.sin(lat1) - math.sin(lat2
        ) * math.cos(lat1) * math.cos(lon1 - lon2)

    bearing2 = math.atan2(y2, x2)

    bearing2 = bearing2 * (180 / math.pi)

    if (bearing2 < 0):
        bearing2 = bearing2 + 360

```

```
    # Elevation angle to rotate the antenna
35  # (Approximation for small distances)

37  a = math.pow((math.sin((lat2 - lat1) / 2)), 2
        ) + math.cos(lat1) * math.cos(lat2) * (
39      math.pow((math.sin((lon2 - lon1) / 2)), 2))
    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))
41  R = 6371000.0
    d = R * c
43
    if (d == 0):
45      elevation1 = 90 * (math.pi / 180)
    else:
47      elevation1 = math.atan((alt2 - alt1) / (d))

49  elevation1 = elevation1 * (180 / math.pi)
    elevation2 = 180 - 90 - elevation1
51
    return elevation1, bearing1, elevation2, bearing2
```

Program F.1 *AntennaTracking function to obtain the angles of elevation and bearing of both antennas, the final angles are in degrees.*

F.2 Algorithm extracting specific parameters from MAVLink packets

```
# Fuction to extract the parameters

# Scales to correct the format of the variables
scale_latlon = 1e-7
scale_hdg = 1e-2
scale_relative_alt = 1e-3

# Function to extract parameters from MAVLink messages
def mavlink_packet(self, m):
    '''handle mavlink packets'''
    state = self

    # Latitude, longitude, heading and altitude of the UAV
    if m.get_type() == 'GLOBAL_POSITION_INT':
        state.lat2 = m.lat * scale_latlon
        state.lon2 = m.lon * scale_latlon
        state.heading2 = m.hdg * scale_hdg

        agl = state.elevation_model.GetElevation(state.lat,
                                                 state.lon)
        if agl is not None:
            state.alt2 = m.relative_alt * scale_relative_alt
                         + state.alt1 - agl
```

Program F.2 *Data handler. Extraction of parameters within MAVLink messages.*

F.3 Algorithm transmitting the desired parameters from the UAV to the GSA

```
import pickle, socket, time

# Function to send whatever we want to send
def sendData(data, interval):
    # Initial parameters of the communication
    ipAdress = "192.168.89.10" #Receiver device's IP
    port = 5005 # Port
    duration = interval #Number of seconds to send the packets

    # Timeout to stop the sending process
    timeout = time.time() + duration
    sent = 0 #Number of packets sent

    # Text to serialize (pickle or son)
    variableToPickle = data

    # Open a new socket
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    # Create packet with the information wanted.
    # Pickle (or json) to serialize it
    toTransmit = pickle.dumps(variableToPickle)

    while(True):
        if (time.time() > timeout):
            break
        else:
            pass

        # Send the packets
        sock.sendto(toTransmit, (ipAdress, port))
        sent = sent + 1
        print("Sent %s packets to %s through
              port 8080 during %s seconds" %(sent, ipAddress,
                                             duration))
```

Program F.3 *Data handler in the UAV, acting as sender.*

F.4 Algorithm handling the movement of the servo motors

```
import Adafruit_BBIO.PWM as PWM, time, math

# Initial parameters for the servo-motors
servoPinV = P9_16, servoPinH = P9_14

# Initial angles to set the servos at the beginning of program
initialAngleV = 150, initialAngleH = 50
previousAngleV = initialAngleV, previousAngleH = initialAngleH

# Initial Ducy Cycle of the previous initial angle
initialDCV = (9./180.)*initialAngleV + 3
initialDCH = (9./180.)*initialAngleH + 3

# Set the PWM signal to provide initial angle
PWM.start(servoPinV, initialDCV, 50)
PWM.start(servoPinH, initialDCH, 50)

# Time to wait from angle to angle (Bigger than 0.02)
velocity = 0.1

# Function to move the servos to the desired angles
def moveServos(desiredAngleV, desiredAngleH):

    if((desiredAngleV == exit) or (desiredAngleH == exit)):
        print(Exiting the program)
        exit()
    if (desiredAngleV > 180 or desiredAngleH > 180):
        print(One angle inputted is greater than 180 degrees)
    elif (desiredAngleV < 0 or desiredAngleH < 0):
        print(One angle inputted is lower than 0 degrees)
    else:
        toRotateV = abs(desiredAngleV - previousAngleV)
        toRotateH = abs(desiredAngleH - previousAngleH)

        stepH = 0, stepV = 0

        if (desiredAngleV > previousAngleV and
            desiredAngleH > previousAngleH):

            while (stepV < toRotateV or stepH < toRotateH):
                if (stepV < toRotateV):
                    stepV = stepV + 1
                    dutyCycleV = (9. / 180.) * (previousAngleV
                    + stepV) + 3
                    PWM.set_duty_cycle(servoPinV, dutyCycleV)
                if (stepH < toRotateH):
```

```
47                      stepH = stepH + 1
                        dutyCycleH = (9. / 180.) * (previousAngleH
49                      + stepH) + 3
                        PWM.set_duty_cycle(servoPinH, dutyCycleH)
51                  time.sleep(velocity)

53          elif (desiredAngleV < previousAngleV and
                  desiredAngleH < previousAngleH):
55
                while (stepV < toRotateV or stepH < toRotateH):
57                  if (stepV < toRotateV):
                        stepV = stepV + 1
59                      dutyCycleV = (9. / 180.) * (previousAngleV
                        - stepV) + 3
61                      PWM.set_duty_cycle(servoPinV, dutyCycleV)
                    if (stepH < toRotateH):
63                      stepH = stepH + 1
                        dutyCycleH = (9. / 180.) * (previousAngleH
65                      - stepH) + 3
                        PWM.set_duty_cycle(servoPinH, dutyCycleH)
67                  time.sleep(velocity)

69          elif (desiredAngleV > previousAngleV and
                  desiredAngleH < previousAngleH):
71
                while (stepV < toRotateV or stepH < toRotateH):
73                  if (stepV < toRotateV):
                        stepV = stepV + 1
75                      dutyCycleV = (9. / 180.) * (previousAngleV
                        + stepV) + 3
77                      PWM.set_duty_cycle(servoPinV, dutyCycleV)
                    if (stepH < toRotateH):
79                      stepH = stepH + 1
                        dutyCycleH = (9. / 180.) * (previousAngleH
81                      - stepH) + 3
                        PWM.set_duty_cycle(servoPinH, dutyCycleH)
83                  time.sleep(velocity)

85          elif (desiredAngleV < previousAngleV and
                  desiredAngleH > previousAngleH):
87
                while (stepV < toRotateV or stepH < toRotateH):
89                  if (stepV < toRotateV):
                        stepV = stepV + 1
91                      dutyCycleV = (9. / 180.) * (previousAngleV
                        - stepV) + 3
93                      PWM.set_duty_cycle(servoPinV, dutyCycleV)
```

```
                if (stepH < toRotateH):
95                  stepH = stepH + 1
                    dutyCycleH = (9. / 180.) * (previousAngleH
97                  + stepH) + 3
                    PWM.set_duty_cycle(servoPinH, dutyCycleH)
99              time.sleep(velocity)

101         elif (desiredAngleV > previousAngleV and
                  desiredAngleH == previousAngleH):
103
            while (stepV < toRotateV):
105             stepV = stepV + 1
                dutyCycleV = (9. / 180.) * (previousAngleV
107             + stepV) + 3
                PWM.set_duty_cycle(servoPinV, dutyCycleV)
109         time.sleep(velocity)

111         elif (desiredAngleV < previousAngleV and
                  desiredAngleH == previousAngleH):
113
            while (stepV < toRotateV):
115             stepV = stepV + 1
                dutyCycleV = (9. / 180.) * (previousAngleV
117             - stepV) + 3
                PWM.set_duty_cycle(servoPinV, dutyCycleV)
119         time.sleep(velocity)

121         elif (desiredAngleV == previousAngleV and
                  desiredAngleH > previousAngleH):
123
            while (stepH < toRotateH):
125             stepH = stepH + 1
                dutyCycleH = (9. / 180.) * (previousAngleH
127             + stepH) + 3
                PWM.set_duty_cycle(servoPinH, dutyCycleH)
129         time.sleep(velocity)

131         elif (desiredAngleV == previousAngleV and
                  desiredAngleH < previousAngleH):
133
            while (stepH < toRotateH):
135             stepH = stepH + 1
                dutyCycleH = (9. / 180.) * (previousAngleH
137             - stepH) + 3
                PWM.set_duty_cycle(servoPinH, dutyCycleH)
139         time.sleep(velocity)
```

```
141         else:
                print(Please insert a valid angle)
143
    # Loop to move servos
145 # In the final code it is implemented differently.
    while(1):
147     desiredAngleV = input(Input vertical angle or exit)
        desiredAngleH = input(Input horizontal angle or exit)
149     moveServos(desiredAngleV, desiredAngleH)
        previousAngleV = desiredAngleV
151     previousAngleH = desiredAngleH
```

Program F.4 *moveServos function to handle the movement of the servo motors to a specific angle. In the final application, this angle will be the elevation or bearing to rotate the antennas.*

F.5 Class definition of the completely new module to be created within MAVProxy

```
class tracking(mp_module.MPModule):
    def __init__(self, mpstate):
        """Initialise module"""
        super(tracking, self).__init__(mpstate,"TrackingSystem",
                                       "Tracking system for the antenna")
        self.add_command('tracking', self.cmd_tracking,
                         "Tracking system for the antenna",
                         ['status', 'set (LOGSETTING)'])

        # Params of the ground station antenna. Set manually
        self.lat1 = 0
        self.lon1 = 0
        self.alt1 = 0
        self.heading1 = 0

        self.elevation1 = 0
        self.bearing1 = 0

        # Params of the drone
        self.lat2 = 0
        self.lon2 = 0
        self.alt2 = 0
        self.heading2 = 0

        self.elevation2 = 0
        self.bearing2 = 0

        self.elevation_model = mp_elevation.ElevationModel()
        self.status_callcount = 0

        self.packets_mytarget = 0
        self.packets_othertarget = 0
        self.verbose = False

        self.tracking_settings = mp_settings.MPSettings(
            [ ('verbose', bool, False),])
```

Program F.5 *Class definition of the new MAVProxy module to be created.*

F.6 Algorithm handling the reception of UDP datagrams in the GS

```
# Python script to execute the RX loop in the ground station
import socket
import pickle
import time
import Adafruit_BBIO.PWM as PWM

# Initial parameters for the servo-motors (Pins in Beagle-Bone)
servoPinV = "P9_16"
servoPinH = "P9_14"

# Initial angles to set the servos when beginning the program
initialAngleV = 150
initialAngleH = 50

# Initial Ducy Cycle of the previous initial angle
initialDCV = (9./180.)*initialAngleV + 3
initialDCH = (9./180.)*initialAngleH + 3

# Set the PWM signal to provide initial angle
PWM.start(servoPinV, initialDCV, 50)
PWM.start(servoPinH, initialDCH, 50)

while (1):
    # Initial parameters of the communication
    ipAdress = "192.168.89.10" # Ip address of the RX host
    port = 5005

    # To receive data during X seconds:
    duration = 7  # Number of seconds to send the packets
    timeout = time.time() + duration

    received = 0 # Number of packets received

    # Open a new socket
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    # Bind the IP address and port
    sock.bind((ipAdress, port))

    # I receive data during 5 seconds
    while(True):
        if (time.time() > timeout):
            break
        else:
         pass
```

```
46          # Receive the packets
            data, addr = sock.recvfrom(1024)
48          received = received + 1
            print("Received %s packets in 192.168.89.10 through
50          port 5005 during %s seconds" %(received, duration))

52      # Deserialized and extraction of useful information
        toRead = pickle.loads(data)
54      elevation1 = data[1]
        bearing1 = data[2]
56
        # Once received the data, I move the servo motors to
58      # the angles received
        if (elevation1 > 180 or bearing1 > 180):
60          print("One angle inputted is greater than 180 degrees")

62      elif (elevation1 < 0 or bearing1 < 0):
            print("One angle inputted is lower than 0 degrees")
64
        else:
66
            # Convertion from angles to duty cycle
68          dutyCycleV = (9. / 180.) * (elevation1) + 3
            dutyCycleH = (9. / 180.) * (bearing1) + 3
70
            # Apply the duty cycle to the servos
72          PWM.set_duty_cycle(servoPinV, dutyCycleV)
            PWM.set_duty_cycle(servoPinH, dutyCycleH)
```

Program F.6 *Ground station's end. Reception of UDP datagrams and movement of the servo motors.*

BIBLIOGRAPHY

[1] P. Kumar, S. Garg, A. Singh, S. Batra, N. Kumar, and I. You, "MVO-based Two-Dimensional Path Planning Scheme for Providing Quality of Service in UAV Environment," *IEEE Internet of Things Journal*, no. 99, pp. 1–1, 2018.

[2] N. Makitalo, A. Ometov, J. Kannisto, S. Andreev, Y. Koucheryavy, and T. Mikkonen, "Safe, Secure Executions at the Network Edge: Coordinating Cloud, Edge, and Fog Computing," *IEEE Software*, no. 1, pp. 30–37, 2017.

[3] G. Wang, K. Lim, B.-S. Lee, and J. Y. Ahn, "Handover key management in an lte-based unmanned aerial vehicle control network," in *2017 IEEE 5th International Conference on Future Internet of Things and Cloud: Workshops (W-FiCloud)*. IEEE, 2017, pp. 200–205.

[4] J. Lyu, Y. Zeng, R. Zhang, and T. J. Lim, "Placement optimization of uav-mounted mobile base stations," *IEEE Communications Letters*, vol. 21, no. 3, pp. 604–607, 2017.

[5] L. Yang, J. Qi, J. Xiao, and X. Yong, "A literature review of uav 3d path planning," in *Intelligent Control and Automation (WCICA), 2014 11th World Congress on*. IEEE, 2014, pp. 2376–2381.

[6] A. Orsino, A. Ometov, G. Fodor, D. Moltchanov, L. Militano, S. Andreev, O. N. Yilmaz, T. Tirronen, J. Torsner, G. Araniti *et al.*, "Effects of Heterogeneous Mobility on D2D-and Drone-Assisted Mission-Critical MTC in 5G," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 79–87, 2017.

[7] M. Alzenad, A. El-Keyi, F. Lagum, and H. Yanikomeroglu, "3d placement of an unmanned aerial vehicle base station (uav-bs) for energy-efficient maximal coverage," *IEEE Wireless Communications Letters*, vol. 6, no. 4, pp. 434–437, 2017.

[8] M. Mahardika, G. Nugroho, and E. Y. Prasetyo, "Uav long range surveillance system based on biquad antenna for the ground control station," in *Research and Development (SCOReD), 2016 IEEE Student Conference on*. IEEE, 2016, pp. 1–5.

[9] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Optimal Transport Theory for Cell Association in UAV-Enabled Cellular Networks," *IEEE Communications Letters*, vol. 21, no. 9, pp. 2053–2056, 2017.

[10] J. Sae, S. F. Yunas, and J. Lempiainen, "Coverage aspects of temporary lap network," in *Wireless On-demand Network Systems and Services (WONS), 2016 12th Annual Conference on*. IEEE, 2016, pp. 1–4.

[11] A. Al-Hourani, S. Kandeepan, and A. Jamalipour, "Modeling air-to-ground path loss for low altitude platforms in urban environments," in *Global Communications Conference (GLOBECOM), 2014 IEEE*. IEEE, 2014, pp. 2898–2904.

[12] M. Helmy, Z. Ankarali, M. Siala, T. Baykas, and H. Arslan, "Dynamic utilization of low-altitude platforms in aerial heterogeneous cellular networks," in *IEEE 18th Wireless and Microwave Technology Conference (WAMICON)*. IEEE, 2017, pp. 1–6.

[13] A. AL-Hourani, S. Chandrasekharan, G. Kaandorp, W. Glenn, A. Jamalipour, and S. Kandeepan, "Coverage and rate analysis of aerial base stations," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 6, pp. 3077–3081, 2016.

[14] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP Altitude for Maximum Coverage," *IEEE Wireless Communications Letters*, vol. 3, no. 6, pp. 569–572, 2017.

[15] A. Al-Hourani, S. Kandeepan, and A. Jamalipour, "Stochastic Geometry Study on Device-to-Device Communication as a Disaster Relief Solution," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3005–3017, 2016.

[16] H. Choi, M. Geeves, B. Alsalam, and F. Gonzalez, "Open source computer-vision based guidance system for uavs on-board decision making," in *Aerospace Conference*. IEEE, 2016, pp. 1–5.

[17] "Mavlink," 2017, [Accessed: 06-11-2017]. [Online]. Available: http://mavlink.org/messages/common

[18] S. Atoev, K.-R. Kwon, S.-H. Lee, and K.-S. Moon, "Data analysis of the mavlink communication protocol," in *Information Science and Communications Technologies (ICISCT)*. IEEE, 2017, pp. 1–3.

[19] Z. Xiao, "An Efficient Power Over Ethernet (PoE) Interface With Current-Balancing and Hot-Swapping Control," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 3, pp. 2496–2506, 2017.

[20] A. Sarhan and S. Qin, "Robust adaptive flight controller for uav systems," in *2017 4th International Conference on Information Science and Control Engineering (ICISCE)*. IEEE, 2017, pp. 1214–1219.

[21] M. Ilamathi and K. Abirami, "Automatic target tracker for main battle tank," in *Communications and Signal Processing (ICCSP), 2015 International Conference on.* IEEE, 2015, pp. 0318–0321.

[22] W. Tan, K. Liu, and J. Liao, "Study of s-band antenna tracking algorithm based on fpga," in *Ubiquitous Wireless Broadband (ICUWB), 2016 IEEE International Conference on.* IEEE, 2016, pp. 1–4.

[23] G. Apaydin and L. Sevgi, *Wave Propagation Over Flat Earth.* Wiley-IEEE Press, 2018, pp. 152–. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=8052378

[24] T. Monawar, S. B. Mahmud, and A. Hira, "Anti-theft vehicle tracking and regaining system with automatic police notifying using Haversine formula," in *International Conference on Advances in Electrical Engineering (ICAEE).* IEEE, 2017, pp. 775–779.

[25] E. Choi and S. Chang, "A consumer tracking estimator for vehicles in GPS-free environments," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 4, pp. 450–458, 2017.

[26] M. V. Gheorghe and M. C. Bodea, "Calibration Optimization Study for Tilt-Compensated Compasses," *IEEE Transactions on Instrumentation and Measurement*, vol. PP, no. 99, pp. 1–9, 2018.

[27] D. E. Bolanakis, "MEMS barometers and barometric altimeters in industrial, medical, aerospace, and consumer applications," *IEEE Instrumentation & Measurement Magazine*, vol. 20, no. 6, pp. 30–55, 2017.

[28] Y. Zou, R. Gu, D. Wang, A. Jiang, and C. H. Ritz, "Learning a robust DOA estimation model with acoustic vector sensor cues," *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 1688–1691, 2017.

[29] E. Greenberg and P. Levy, "Channel characteristics of uav to ground links over multipath urban environments," in *Microwaves, Antennas, Communications and Electronic Systems (COMCAS), 2017 IEEE International Conference on.* IEEE, 2017, pp. 1–4.

[30] W. Khawaja, I. Guvenc, and D. Matolak, "Uwb channel sounding and modeling for uav air-to-ground propagation channels," in *2016 IEEE Global Communications Conference (GLOBECOM).* IEEE, 2016, pp. 1–7.

[31] A. M. Hayajneh, S. A. Raza Zaidi, D. C. McLernon, and M. Ghogho, "Optimal dimensioning and performance analysis of drone-based wireless communications," in *Globecom Workshops (GC Wkshps)*. IEEE, 2016, pp. 1–6.

[32] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Performance optimization for uav-enabled wireless communications under flight time constraints," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[33] G. Strupka, A. Levchenkov, and M. Gorobetz, "Fuzzy-logic algorithm of uav hardware configuration assessment for flight time and lift capacity improvements," in *Power and Electrical Engineering of Riga Technical University (RTUCON), 2017 IEEE 58th International Scientific Conference on*. IEEE, 2017, pp. 1–5.

[34] L. Klepac and D. Rozehnal, "Scavenging simulation of small two-stroke engine with low-pressure fuel injection for usage in unmanned aerial vehicle (uav)," in *International Conference on Military Technologies (ICMT)*. IEEE, 2017, pp. 457–461.

[35] Y. Xu, X. Xia, K. Xu, and WangYurong, "Three-Dimension Massive MIMO for Air-to-Ground Transmission: Location-Assisted Precoding and Impact of AoD Uncertainty," *IEEE Access*, vol. 5, pp. 15 582–15 596, 2017.

[36] J.-U. Park, B.-H. Bae, J.-W. Lee, and J.-H. Kim, "Design of failsafe architecture for unmanned ground vehicle," in *Control Automation and Systems (ICCAS), 2010 International Conference on*. IEEE, 2010, pp. 1101–1104.

[37] B. C. Min, E. T. Matson, and J. W. Jung, "Active antenna tracking system with directional antennas for enhancing wireless communication capabilities of a networked robotic system," *Journal of Field Robotics*, vol. 33, no. 3, pp. 391–406, 2016.

[38] G. Fuggetti, A. Ghetti, and M. Zanzi, "Safety improvement of fixed wing mini-uav based on handy fdi current sensor and a failsafe configuration of control surface actuators," in *Metrology for Aerospace (MetroAeroSpace)*. IEEE, 2015, pp. 356–361.

[39] "Lm2575 adj," 2017, [Accessed: 23-11-2017]. [Online]. Available: http://www.ti.com/lit/ds/symlink/lm2575-n.pdf